AD-A071 076 NAVAL POSTGRADUATE SCHOOL MONTEREY CA F/G 14/2
COMPENSATOR OPTIMIZATION IN MULITPLE INPUTMULTIPLE OUTPUT CONTR--ETC(U)
MAR 79 J T MOWREY

UNCLASSIFIED NL

1 OF 1
AD
A071076

END
DATE
FILMED
8 -79
DDC

(2) LEVEL II

AD A071076

NAVAL POSTGRADUATE SCHOOL
Monterey, California

D D C
RECEIVED
JUL 11 1979
B

(9) Masters THESIS

(6) Compensator Optimization in Multiple Input Multiple Output Control Systems.

by

(11) March 1979

(10) John Tinney Mowrey (12) 80p.

March 1979

Thesis Advisor: H. A. Titus

DDC FILE COPY

Approved for public release; distribution unlimited

251 450

79 07 11 039

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Compensator Optimization in Multiple Input Multiple Output Control Systems		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis; March 1979
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) John Tinney Mowrey, Major, USMC		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93940		12. REPORT DATE March 1979
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Naval Postgraduate School Monterey, CA 93940		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Control systems
Transfer Matrix Method
Cost Function
Free parameter constraints
Multivariable Systems, Linear
Transfer Function Application

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The BOXPLX of a user specified cost function is evaluated as a control system design method. The emphasis is on compensator design for multiple-input-multiple-output plants, but the technique should be applicable to any control system with adjustable parameters.

The engineer must define the plant dynamics, inputs and

over

DD FORM 1473 1 JAN 73 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601 |

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

outputs, the desired output responses for the specified inputs, weighting factors for the performance measure, the type of compensation to be used, the free parameters, and initial values and constraints on the free parameters.

The program gives a direct search of the optimum values, within the specified constraints, for the free parameters. Evaluation of the results is given in graphical plots of the output time responses including the desired output response as well as the compensated response.

The Transfer Matrix Method is used to provide an analytical check on the accuracy of the method and the procedure is illustrated with a two-input-two-output system example.

Approved for public release; distribution unlimited

COMPENSATOR OPTIMIZATION IN MULTIPLE INPUT MULTIPLE OUTPUT CONTROL SYSTEMS

by

John Tinney Mowrey
Major, United States Marine Corps
B.S.E.E., University of Florida, 1963

Submitted in partial fulfillment of the
requirement for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL

March 1979

Author: John T. Mowrey

Approved by: Thesis Advisor

Second Reader

Chairman, Department of Electrical Engineering

Dean of Science and Engineering

ABSTRACT

The BOXPLX of a user specified cost function is evaluated as a control system design method. The emphasis is on compensator design for multiple-input-multiple-output plants, but the technique should be applicable to any control system with adjustable parameters.

The engineer must define the plant dynamics, inputs and outputs, the desired output responses for the specified inputs, weighting factors for the performance measure, the type of compensation to be used, the free parameters, and initial values and constraints on the free parameters.

The program gives a direct search of the optimum values, within the specified constraints, for the free parameters. Evaluation of the results is given in graphical plots of the output time responses including the desired output response as well as the compensated response.

The Transfer Matrix Method is used to provide an analytical check on the accuracy of the method and the procedure is illustrated with a two-input-two-output system example.

TABLE OF CONTENTS

LIST OF FIGURES

I. INTRODUCTION

As control systems become more and more complex, design engineers are faced with an increasingly difficult problem in the development of compensators. Single-input-single-output linear systems can be compensated using Bode plots or root locus plots and trial and error methods to meet the system specifications [1]. When compensating multiple-input-multiple-output linear systems, use can be made of transfer matrix techniques described by Ogata [2] to achieve total decoupling and the desired output responses. Since decoupling during the transient period of real systems may require controls which exceed physical constraints imposed by the system, it is sometimes desirable to decouple during steady state conditions and allow coupling during the transient period [3].

Non-linear, single-input, single-ouput systems, wherein the non-linearity can be lumped into one element, can be analyzed and compensated effectively using the phase plane method (for second order systems) and describing function analysis [2,4]. Some of these methods require extensive mathematical manipulations and become rather unwieldy, but do produce the desired results. In all of the analysis and design approaches mentioned above, any one of several well established simulation programs can be used to evaluate the design.

To reduce the tedium involved in the design of compensators, frequency domain and time domain computer optimization schemes have been developed, which will set free parameters in the compensator to yield a system response which most nearly

approximates a reference response specified by the user. Lima used this approach to address the problem of controller design for ships engaged in underway replenishment [5]. MacNamara applied this method to an autopilot design [6], and Vines developed a generalized program for compensator optimization in single-input-single-output systems [7].

The following chapters present a generalized discussion of transfer matrix analysis and compensation of multivariable systems, optimization techniques, and performance measures. A two-input-two-output system is then compensated using transfer matrix techniques, and using parameter optimization. The results are then compared.

II. MULTIVARIABLE SYSTEM COMPENSATION

A. TRANSFER MATRIX-ANALYTICAL METHODS

1. Introduction

A linear, multiple-input-multiple-output or multivariable system can be described by a transfer matrix, which is simply an extension of the transfer function concept. Consider the two-input-two-output open loop plant:

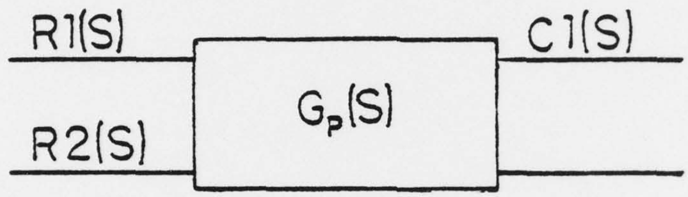

Figure 2-1 Multivariable Plant

The Block Diagram of the plant can be represented in the general sense as shown in Figure 2-2.

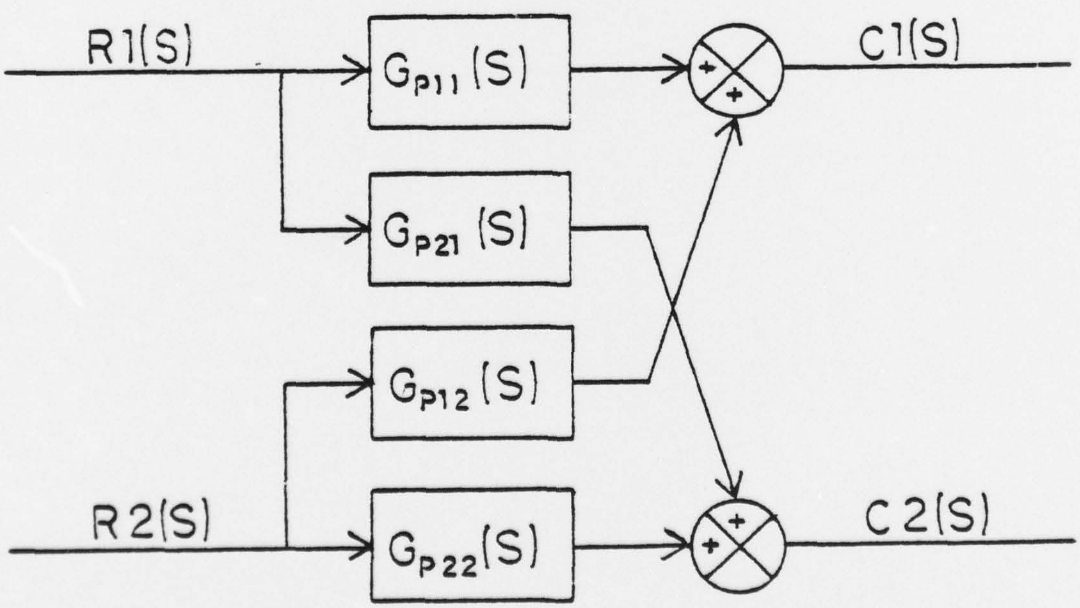

Figure 2-2 Generalized Multivariable Plant

Algebraic equations for the outputs in the independent variable S, can be written directly:

$$C1(s) = Gp11(s)R1(s) + Gp12(s)\ R2(s) \tag{1}$$

$$C2(s) = Gp21(s)R1(s) + Gp22(s)\ R2(s) \tag{2}$$

If the inputs $Rj(s)$ and the outputs $C_i(s)$ are each considered as vectors of one dimension ($i, j = 1, 2$) the equations can be written in matrix form:

$$\begin{bmatrix} C1(s) \\ C2(s) \end{bmatrix} = \begin{bmatrix} Gp11(s) & Gp12(s) \\ Gp21(s) & Gp22(s) \end{bmatrix} \begin{bmatrix} R1(s) \\ R2(s) \end{bmatrix} \tag{3}$$

The number of inputs and outputs can be increased, and need not be equal so that in the general case:

$$\begin{bmatrix} C1(s) \\ C2(s) \\ 1 \\ 1 \\ 1 \\ C_i(s) \end{bmatrix} = \begin{bmatrix} Gp11(s) & Gp12(s) & --- & Gpij(s) \\ Gp21(s) & & & \\ 1 & & & \\ 1 & & & \\ 1 & & & \\ Gpi1(s) & & & Gpij(s) \end{bmatrix} \begin{bmatrix} R1(s) \\ R2(s) \\ \\ \\ \\ R_j(s) \end{bmatrix} \tag{4}$$

or:

$$\underset{\sim}{C}(s) = \underset{\sim}{G}_p(s)\underset{\sim}{R}(s) \tag{5}$$

where $Gp(s)$ is the transfer matrix $(i \times j)$ of the plant.

2. Feedback Compensation

Transfer matrices can be used to describe the compensation of multivariable systems as well as the plant. Consider the two-input-two-ouput plant above with feedback compensation

as shown in Figure 2-3.

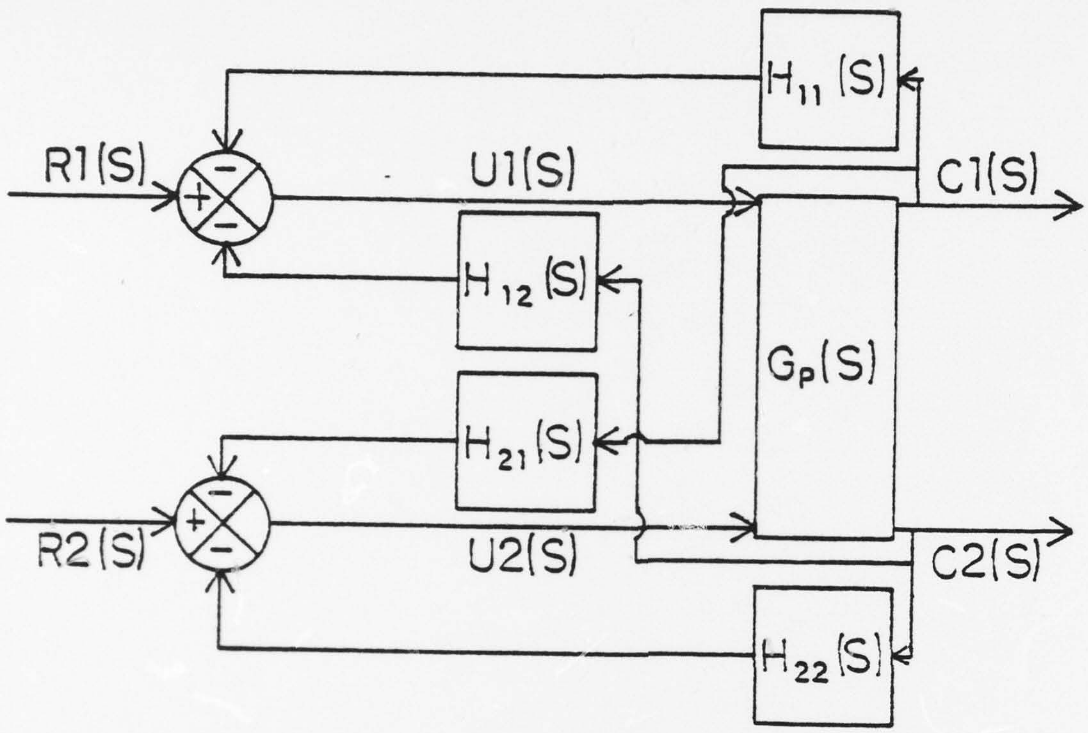

Figure 2-3

Multivariable Plant with Feedback Compensation

Algebraic equations for $R_1(s)$ and $R_2(s)$ can be written:

$$R1(s) = U1(s) - H\,11(s)\,C1(s) - H12(s)\,C2(s) \tag{6}$$

$$R2(s) = U2(s) - H\,21(s)\,C1(s) - H22(s)\,C2(s) \tag{7}$$

with C, R, and U two element, one dimension vectors these equations can be rewritten:

$$\begin{bmatrix} R1(s) \\ R2(s) \end{bmatrix} = \begin{bmatrix} U1(s) \\ U2(s) \end{bmatrix} - \begin{bmatrix} H11(s) & H12(s) \\ H21(s) & H22(s) \end{bmatrix} \begin{bmatrix} C1(s) \\ C2(s) \end{bmatrix} \tag{8}$$

or in the general case:

$$\underset{\sim}{R}(s) = \underset{\sim}{U}(s) - \underset{\sim}{H}(s)\underset{\sim}{C}(s) \qquad (9)$$

where H(s) is the transfer matrix (jxi) of the compensation.

From the uncompensated plant:

$$\underset{\sim}{C}(s) = \underset{\sim}{G}_p(s)\underset{\sim}{R}(s) \qquad (10)$$

Substituting equation (9) for R(s) in Equation (10):

$$\underset{\sim}{C}(s) = \underset{\sim}{G}_p(s) \; [\, \underset{\sim}{U}(s) - \underset{\sim}{H}(s) \; \underset{\sim}{C}(s)] \qquad (11)$$

Rearranging:

$$[\underset{\sim}{I} + \underset{\sim}{G}_p(s)\underset{\sim}{H}(s)]\underset{\sim}{C}(s) = \underset{\sim}{G}_p(s)\underset{\sim}{U}(s) \qquad (12)$$

Since Gp(s) is an ixj matrix, H(s) is a jxi matrix the product, Gp(s)H(s), is a square, ixi matrix. Assuming that $[\underset{\sim}{I} + \underset{\sim}{Gp}(s)\underset{\sim}{H}(s)]$ is *non-singular* the inverse can be taken.

Premultiplying both sides of equation (12) by $[\underset{\sim}{I}+\underset{\sim}{G}_p(s)\underset{\sim}{H}(s)]^{-1}$:

$$\underset{\sim}{C}(s) = [\underset{\sim}{I} + \underset{\sim}{G}_p(s)\underset{\sim}{H}(s)]^{-1}\underset{\sim}{G}_p(s)\underset{\sim}{U}(s) \qquad (13)$$

$$\underset{\sim}{C}(s) = \underset{\sim}{G}(s)\underset{\sim}{U}(s) \qquad (14)$$

The compensated system transfer matrix (ixj) is then:

$$\underset{\sim}{G}(s) = [\underset{\sim}{I} + \underset{\sim}{G}_p(s)\underset{\sim}{H}(s)]^{-1}\underset{\sim}{G}_p(s) \qquad (15)$$

The plant with feedback compensation can be depicted more clearly as shown in Figure 2-4.

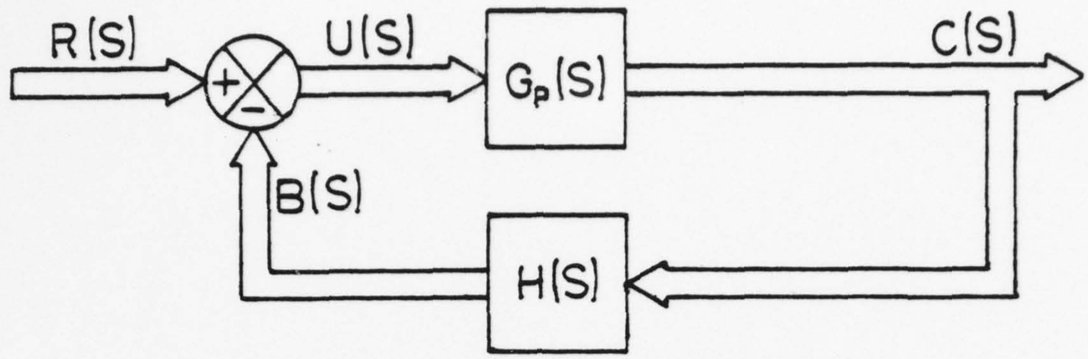

Figure 2-4

Matrix/Vector Representation of Multivariable Plant with Feedback Compensation

To determine the feedback compensation, $\underset{\sim}{H}(s)$, required to achieve design specifications, the dynamics of the plant must be known and put into transfer matrix form, $\underset{\sim}{G}_p(s)$. The desired closed loop dynamics must then be reflected in the closed loop transfer matrix, $\underset{\sim}{G}(s)$. Then through matrix manipulation the compensation, $\underset{\sim}{H}(s)$, can be computed in closed form provided that $\underset{\sim}{G}(s)$ and $\underset{\sim}{G}_p(s)$ are square and non-singular.

Premultiplying both sides of equation (15) by $[\underset{\sim}{I}+\underset{\sim}{G}_p(s)\underset{\sim}{H}(s)]$ and then postmultiplying by $\underset{\sim}{G}(s)^{-1}$:

$$\underset{\sim}{I} + \underset{\sim}{G}_p(s)\underset{\sim}{H}(s) = \underset{\sim}{G}_p(s)\underset{\sim}{G}(s)^{-1} \tag{16}$$

Rearranging:

$$\underset{\sim}{G}_p(s)\underset{\sim}{H}(s) = \underset{\sim}{G}_p(s)\underset{\sim}{G}(s)^{-1} - \underset{\sim}{I} \tag{17}$$

Then multiplying both sides of equation (16) by $G_p(s)^{-1}$

$$\underset{\sim}{H}(s) = \underset{\sim}{G}(s)^{-1} - \underset{\sim}{G}_p(s)^{-1} \tag{18}$$

3. Cascade Compensation

Cascade compensation can be treated in a similar fashion. Consider the same basic plant with cascade compensation as shown

in Figure 2-5.

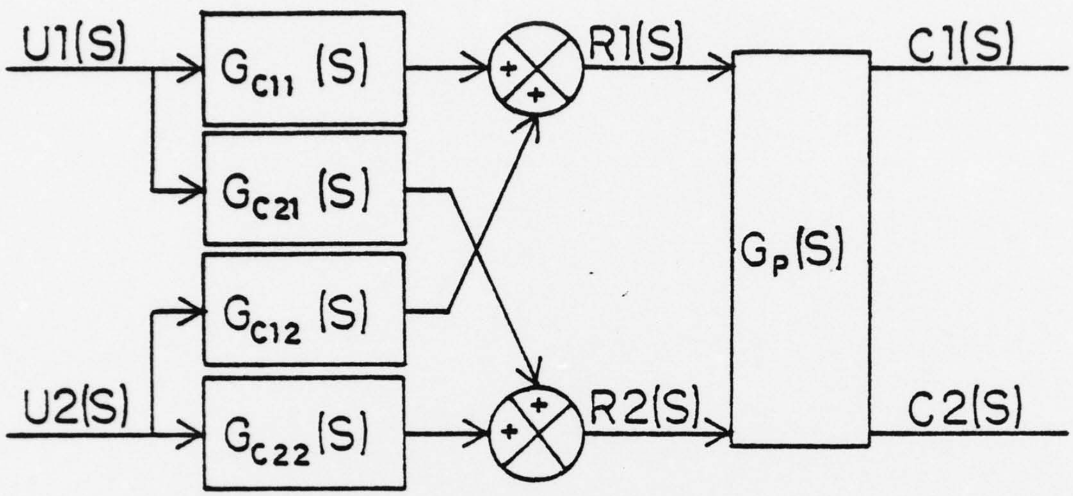

Figure 2-5

Multivariable Plant with Cascade Compensation

The algebraic equations for $R_1(s)$ and $R_2(s)$ are:

$$R1(s) = Gc11(s)U_1(s) + Gc12(s)U2(s) \tag{19}$$

$$R2(s) = Gc21(s)U1(s) + Gc22(s)U2(s) \tag{20}$$

or
$$\begin{bmatrix} R1(s) \\ R2(s) \end{bmatrix} = \begin{bmatrix} Gc11(s) & Gc12(s) \\ Gc21(s) & Gc22(s) \end{bmatrix} \begin{bmatrix} U1(s) \\ U2(s) \end{bmatrix} \tag{21}$$

in the general case:

$$\underset{\sim}{C}(s) = \underset{\sim}{G}p(s)\underset{\sim}{G}c(s)\underset{\sim}{U}(s) \tag{22}$$

$$\underset{\sim}{G}(s) = \underset{\sim}{G}p(s)\underset{\sim}{G}c(s) \tag{23}$$

If $\underset{\sim}{\underset{\sim}{Gp}}(s)$ is a square non-singular matrix the compensation $\underset{\sim}{\underset{\sim}{Gc}}(s)$ is given by:

$$\underset{\sim}{G}c(s) = \underset{\sim}{G}p(s)^{-1}\ \underset{\sim}{G}(s) \tag{24}$$

The compensated plant can be depicted as shown in Figure 2-6.

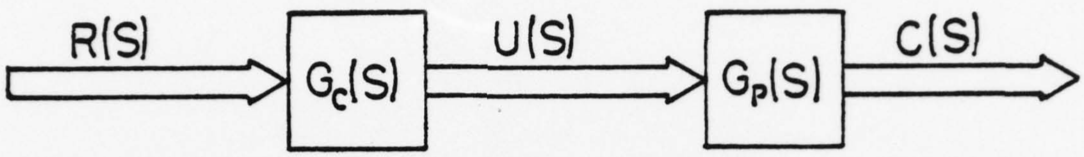

Figure 2-6

Matrix/Vector Representation of Multivariable Plant with Cascade Compensation

4. Cascade and Feedback Compensation.

Cascade and feedback compensation can also be combined as shown in Figure 2-7.

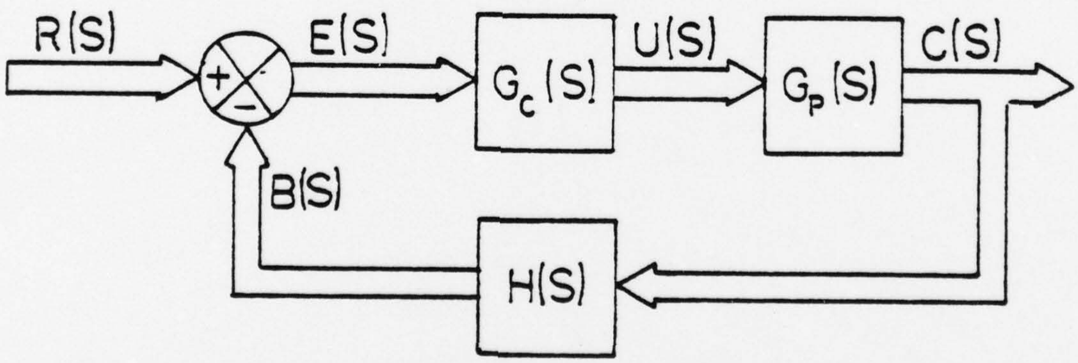

Figure 2-7

Matrix/Vector Representation of Multivariable Plant with Cascade and Feedback Compensation

$$\underset{\sim}{E}(s) = \underset{\sim}{R}(s) - \underset{\sim}{H}(s)\underset{\sim}{C}(s) \tag{25}$$

$$\underset{\sim}{U}(s) = \underset{\sim}{G}_c(s)\underset{\sim}{E}(s) \tag{26}$$

$$\underset{\sim}{C}(s) = \underset{\sim}{G}_p(s)\underset{\sim}{U}(s) = \underset{\sim}{Gp}(s)\underset{\sim}{Gc}(s)\ [\underset{\sim}{R}(s) - \underset{\sim}{H}(s)\underset{\sim}{C}(s)] \tag{27}$$

$$\underset{\sim}{C}(s) = \underset{\sim}{Gp}(s)\underset{\sim}{Gc}(s)\underset{\sim}{R}(s) - \underset{\sim}{Gp}(s)\underset{\sim}{Gc}(s)\underset{\sim}{H}(s)\underset{\sim}{C}(s) \tag{28}$$

$$\underset{\sim}{Gp}(s)\underset{\sim}{Gc}(s)\underset{\sim}{R}(s) = [\underset{\sim}{I} + \underset{\sim}{Gp}(s)\underset{\sim}{Gc}(s)\underset{\sim}{H}(s)]\underset{\sim}{C}(s) \tag{29}$$

If Gp(s) is an ixj matrix Gc(s) will be a j x j matrix, and H(s) will be jxi, in which case Gp(s)Gc(s)H(s) is a square matrix (ixi). If $[\underset{\sim}{I} + \underset{\sim}{Gp}(s)\underset{\sim}{Gc}(s)\underset{\sim}{H}(s)]$ is non-singular, it can be inverted and:

$$\underset{\sim}{C}(s) = [\underset{\sim}{I} + \underset{\sim}{Gp}(s)\underset{\sim}{Gc}(s)\underset{\sim}{H}(s)]^{-1}\underset{\sim}{Gp}(s)\underset{\sim}{Gc}(s)\underset{\sim}{R}(s) \tag{30}$$

$$\underset{\sim}{C}(s) = \underset{\sim}{G}(s)\ \underset{\sim}{R}(s)$$

The compensated system transfer matrix $\underset{\sim}{G}(\underset{\sim}{s})$ (ixj) is:

$$\underset{\sim}{G}(s) = [\underset{\sim}{I} + \underset{\sim}{Gp}(s)\underset{\sim}{GC}(s)\underset{\sim}{H}(s)]^{-1}\ \underset{\sim}{Gp}(s)\underset{\sim}{Gc}(s) \tag{31}$$

Equation (31) contains two unknown matrices, $\underset{\sim}{H}(s)$ and $\underset{\sim}{Gc}(s)$. One of these must be selected arbitrarily and then in certain special cases the other can be solved explicitly. For example:

$$\underset{\sim}{H}(s) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \underset{\sim}{I} \tag{32}$$

In which case:

$$\underset{\sim}{G}(s) = [\underset{\sim}{I} + \underset{\sim}{Gp}(s)\underset{\sim}{Gc}(s)]^{-1}\ \underset{\sim}{Gp}(s)\underset{\sim}{Gc}(s) \tag{33}$$

$$[\underset{\sim}{I} + \underset{\sim}{Gp}(s)\underset{\sim}{Gc}(s)]\underset{\sim}{G}(s) = \underset{\sim}{Gp}(s)\underset{\sim}{Gc}(s) \tag{34}$$

$$\underset{\sim}{G}(s) = \underset{\sim}{Gp}(s)\underset{\sim}{Gc}(s)\ [\underset{\sim}{I} - \underset{\sim}{G}(s)] \tag{35}$$

If Gp(s) is a square matrix G(s) and Gc(s) will be square. If $\underset{\sim}{Gp}(s)\ [\underset{\sim}{G}(s)^{-1} - \underset{\sim}{I}]$, and $\underset{\sim}{G}(s)$ are non-singular:

$$\underset{\sim}{G}p(s)^{-1} = \underset{\sim}{G}c(s)[\underset{\sim}{G}(s)^{-1} - \underset{\sim}{I}] \tag{36}$$

$$\underset{\sim}{G}c(s) = \underset{\sim}{G}p(s)^{-1} [\underset{\sim}{G}(s)^{-1} - \underset{\sim}{I}]^{-1} \tag{37}$$

If $\underset{\sim}{G}c(s)$ and $\{[\underset{\sim}{G}(s)^{-1} - \underset{\sim}{I}]\underset{\sim}{G}p(s)\}$ are assumed to be non-singular equation (37) can be rearranged for convenience:

$$\underset{\sim}{G}p(s)\underset{\sim}{G}c(s) = [\underset{\sim}{G}(s)^{-1} - \underset{\sim}{I}]^{-1} \tag{38}$$

$$\underset{\sim}{G}p(s) = [\underset{\sim}{G}(s)^{-1} - \underset{\sim}{I}]^{-1} \underset{\sim}{G}c(s)^{-1} \tag{39}$$

$$\underset{\sim}{G}c(s)^{-1} = [\underset{\sim}{G}(s)^{-1} - \underset{\sim}{I}] \underset{\sim}{G}p(s) \tag{40}$$

$$\underset{\sim}{G}c(s) = \{[\underset{\sim}{G}(s)^{-1} - \underset{\sim}{I}] \underset{\sim}{G}p(s)\}^{-1} \tag{41}$$

This result can also be obtained directly from equation (37) using the matrix identity $(\underset{\sim}{A}\underset{\sim}{B})^{-1} = \underset{\sim}{B}^{-1} \underset{\sim}{A}^{-1}$.

In each of the systems discussed above the compensation cannot be solved for in closed form, except in special cases as indicated.

B. PARAMETER OPTIMIZATION - COMPUTER METHOD

1. Optimization Technique

The Complex Method of constrained optimization (BoxPLX) was utilized in this program. The algorithm suggested by Box (9) was modified by the programmer, R. R. Hilleary, Naval Postgraduate School, to find the constrained minimum, and includes an integer programming option, as suggested in (10). The Complex Method is a modification of the unconstrained, direct search, Simplex Method introduced in (11). Direct search methods compare function and constraint values only in searching for the minimum value of the function. These direct search methods have been widely used because of their robustness, reliability, and ease in programming and use. They have widespread applicability. The one drawback is that they are generally less efficient than gradiant-based techniques (12). A more detailed discussion of the Complex Method is included in the program documentation.

In this program, the main program simulates the reference output and then calls FUNCTION BOXPLX which in turn calls FUNCTION FE. FUNCTION FE calls SUBROUTINE Plant which simulates the compensated system. FE then computes the performance measure and returns to BOXPLX. BOXPLX keeps track of the compensator parameter values, computes a new set of free parameters and calls FE again. This iteration continues until termination as described in the documentation. See Appendix A for more information on the program.

2. Performance Measures

In order for a numerical method for parameter optimization to produce a result, a decision process must be completed, the result of which is the set of parameters which is defined as "optimum." This term optimum is an enigima of sorts. The program addressed here utilizes a comparison of time domain response of a compensated system with a reference response specified by the user. The word optimum above is dependent on the designer's knowledge of what reference response is best suited to the output he is trying to control. Furthermore, with the exception of very simple linear systems, it is difficult to match a reference response exactly, so the designer must decide if the comparison at steady state is more or less important than that during the transient period. Additionally very high frequencies with periods less than one half the integration step size may be filtered out by the digital process in the computer, and not be present in the simulated output. The designer is faced with some decisions which he must make objectively before the "optimum" set of parameters can be computed.

The decision process mentioned above is based upon a performance measure which is an indicator of the "goodness" of a given set of parameters being varied in the optimization process. The designer must select the performance measure; the performance measure is then minimized by numerical methods, since the objective here is to have the system response match the reference response.

The term selectivity is used to describe a quality of the performance measures. If three dimensional space is considered with the performance measure plotted as a function of two free variables in a compensator of the system, the resultant topology should include one or more minima for the performance measure. The minima with the lowest value are referred to as the global minima, although the minimization performed is constrained. Selectivity is directly related to the steepness of the slope of the contour in the area immediately adjacent to the minima. This selectivity is a function of the performance measure used as well as the plant and compensator dynamics (2,8).

Several performance measures have been suggested in numerous sources, some of which will be discussed here. In each case the performance measure will be represented by the variable J.

Consider a single-input single-output system with output, c(t), and reference response, r(t). The error, e(t), for this case is defined as the difference between the reference response and the system response or:

$$e(t) = r(t) - c(t) \tag{42}$$

Integral square-error. The defining equation for finite time is:

$$J = \int_0^t e^2(t)dt \tag{43}$$

This performance measure is not very selective and tends to emphasize large errors and ignore small ones (2,8).

Integral of time multiplied square-error. The defining equation for finite time is:

$$J = \int_0^t t\, e^2(t)dt \tag{44}$$

Steady state error is penalized more heavily by this performance measure. This performance measure is said to be more selective than the integral square-error criterion in that the value of the integral tends to change more rapidly with changes in the system parameters (2,8).

Integral absolute error. The defining equation for finite time is:

$$J = \int_0^t |e(t)|dt \tag{45}$$

This performance measure is slightly more selective than the integral square-error criterion (2,8).

Integral of time-multiplied absolute error. The defining equation for finite time is:

$$J = \int_0^t t|e(t)|dt \tag{46}$$

This performance measure is more selective than those mentioned previously and, as in the case with the integral of time multiplied square-error criterion, this performance measure tends to emphasize errors late in the transient period and in steady state and deemphasize those which occur early in the transient period (2,8).

The optimization of the performance measure can be addressed in two categories; the case where the reference response specified is the forcing function; or the reference response is some function, usually a second order response, which the designer believes to be a model response for his application. In the first case the peak overshoot, setting time, rise time, and natural frequency, are determined by the dynamics of the compensated system <u>and</u> by the performance measure selected. However, if the designer desires a second order response with a certain minimal rise time, he can specify a second order reference response which meets this requirement, and use the integral absolute error or integral error-squared performance criteria to optimize the compensation parameters.

Provisions for both of these approaches have been included in the program in that the user can reference a second order response, where he specifies the damping factor, natural frequency and gain, or he can utilize the table loop-up feature in which he can specify any reference response he desires.

Other performance measures can be developed using classical measures of system performance such as maximum overshoot, time for the error to reach its first zero, time to reach maximum overshoot, settling time, frequency of oscillation of the transient or steady state error. However, performance measures based on these characteristics would be complex and increase computation time, and their desirability over those previously discussed is questionable.

The performance measure used in the optimization program

(see Appendix) is the sum of the weighted performance measures of each output of the multivariable system.

$$J = \sum_{i=1}^{M} (W_i)(J_{ik}) \tag{47}$$

The subscript i denotes the output for which $J_{i,k}$ is calculated. The subscript k denotes one of four performance measures which can be selected by the user to be applied to each output in turn:

Integral square error

$$Ji,1 = \sum_{j=i}^{N} (Rj,i - C_{j,i})^2 \tag{48}$$

Integral of time multiplied square error

$$Ji,2 = \sum_{j=1}^{N} ((Rj,i - C_{j,i})^2)(i\Delta T) \tag{49}$$

Integral of absolute-error

$$Ji,3 = \sum_{j=1}^{N} | Rj,i - C_{j,i} | \tag{50}$$

Integral of time multiplied square-error

$$Ji,4 = \sum_{j=1}^{N} | Rj,i - C_{j,i}| (i\Delta T) \tag{51}$$

N represents the number of integration steps in the simulation, ΔT represents the integration step size, M is the number of outputs being optimized, and W is the weighting factor applied

to each output.

It should be noted that the performance measures are an approximation and when the weighting factor is introduced the performance measure diverges significantly from the defining integral equation. Comparison of performance measures for a given plant with various types of compensation should be done with the same weighting factors. The same performance measure could be achieved for a system using two different compensation schemes, since the performance measures for each channel may vary significantly.

C. A TWO INPUT-TWO OUTPUT SYSTEM EXAMPLE

1. Transfer Matrix Solution

In order to demonstrate the program it was decided to analyze a simple system, synthesize the compensation required to produce a specified output, and then show that the optimization program will arrive at the same results. It is emphasized that this program will not determine the type of compensation required to achieve the desired response. It will set free parameters in the compensators (specified by the user) to most nearly produce the desired response.

The analysis and compensation was performed using the transfer matrix method described by OGATA (2). The uncompensated plant and the compensated plant were simulated to show that the compensation achieved the design specifications. Certain parameters in the compensator were defined as free parameters. These free parameters were offset from their optimal values and constraints were introduced which restricted the range of values for these parameters during optimization. The plant and compensators with free parameters were simulated in the optimization program and optimization was accomplished.

To illustrate, a simple two-input two-output linear system shown in Figure 3-1 was selected as the plant to be compensated. The design specifications chosen were:

1. Channel one and channel two are to be decoupled during the transient period as well as in steady state.
2. Channel one is to have a second order response to a step input with a natural frequency, W_n, of 10, a

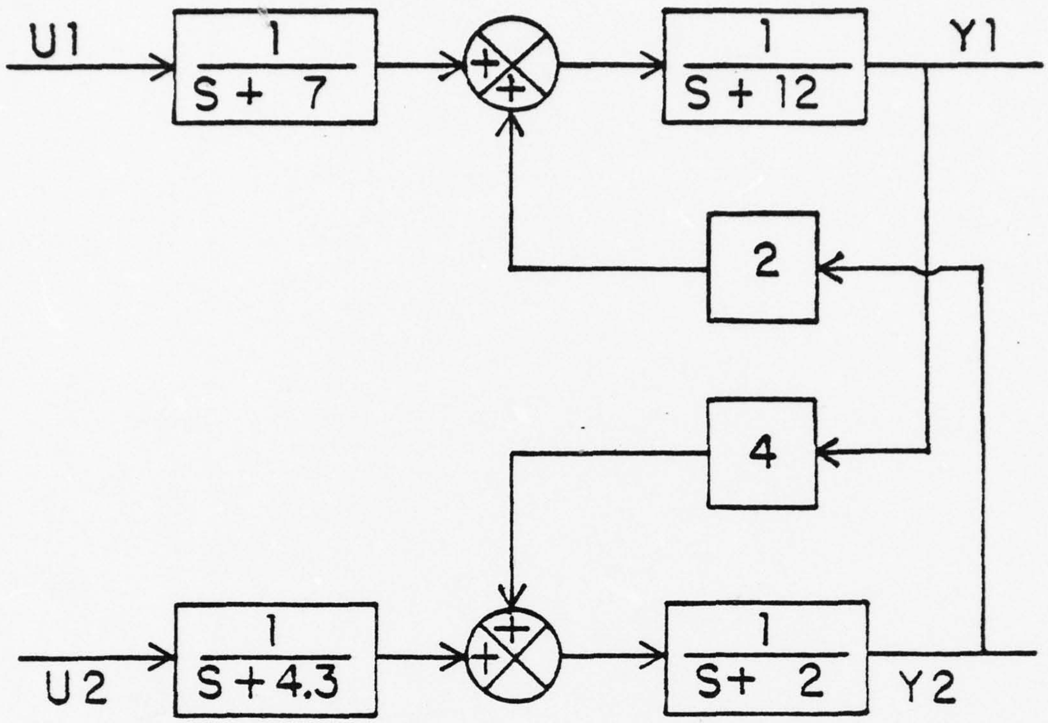

Figure 3-1 Multivariable Plant to be Compensated

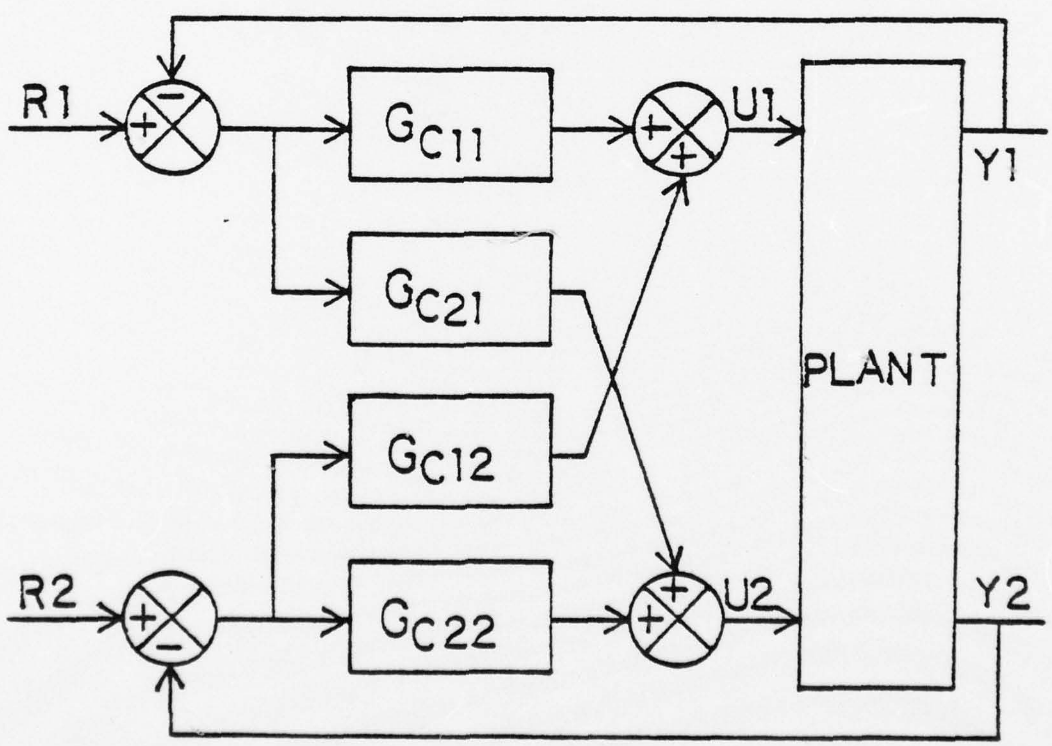

Figure 3-2 Multivariable Plant with Generalized Compensation

damping factor, ζ, of 0.4, and no steady state error.

3. Channel two is to have a second order response to a step input with a natural frequency, W_n, of 4, a damping factor, ζ, of 0.6, and no steady state error.

To achieve total decoupling the off-diagonal elements of the transfer matrix of the compensated system must be zero. The desired compensated system transfer matrix, $\underset{\sim}{G}(\underset{\sim}{s})$, is:

$$G(s) = \begin{bmatrix} \dfrac{100}{S^2 + 8S + 100} & 0 \\ 0 & \dfrac{16}{S^2 + 4.8S + 16} \end{bmatrix} \quad (52)$$

The input/output equations obtained from Figure 3-1 are:

$$Y_1(s) = U_1(s) \frac{1}{(S+7)(S+12)} + Y_2(s) \frac{2}{(s+12)} \quad (53)$$

$$Y_2(s) = U_2(s) \frac{1}{(s+2)(s+4.3)} + Y_1(s) \frac{4}{(s+2)} \quad (54)$$

Substitution of equation (53) into equation (54), and equation (54) into equation (53) produces equations (55) and (56).

$$Y_1(s) = \frac{U_1(s)}{(s+7)(s+12)} + \frac{2U_2(s)}{(s+2)(s+4.3)(s+12)} + \frac{8Y_1(s)}{(s+2)(s+12)} \quad (55)$$

$$Y_2(s) = \frac{U_2(s)}{(s+2)(s+4.3)} + \frac{4U_1(s)}{(s+12)(s+7)(s+2)} + \frac{8Y_2(s)}{(s+2)(s+12)} \quad (56)$$

Collecting like terms in equations (55) and (56) and rearranging:

$$Y_1(s)\ \frac{s^2+14s+16}{(s+2)(s+12)} = \frac{U_1(s)}{(s+7)(s+12)} + \frac{2U_2(s)}{(s+2)(s+4.3)(s+12)} \qquad (57)$$

$$Y_2(s)\ \frac{s^2+14s+16}{(s+2)(s+12)} = \frac{4U_1(s)}{(s+12)(s+7)(s+2)} + \frac{U_2(s)}{(s+2)(s+4.3)} \qquad (58)$$

Multiplying both sides of equations (57) and (58) by

$$\frac{(s+2)(s+12)}{s^2+14s+16}$$

$$Y_1(s) = \frac{(s+2)U_1(s)}{(s+7)(s^2+14s+16)} + \frac{2U_2(s)}{(s+4.3)(s^2+14s+16)} \qquad (59)$$

$$Y_2(s) = \frac{4U_1(s)}{(s+7)(s^2+14s+16)} + \frac{(s+12)U_2(s)}{(s+4.3)(s^2+14s+16)} \qquad (60)$$

The transfer Matrix for the uncompensated plant is:

$$G_p(s) = \begin{bmatrix} \dfrac{(s+2)}{(s+7)(s^2+14s+16)} & \dfrac{2}{(s+4.3)(s^2+14s+16)} \\[2ex] \dfrac{4}{(s+7)(s^2+14s+16)} & \dfrac{(s+12)}{(s+4.3)(2^2+14s+16)} \end{bmatrix} \qquad (61)$$

The generalized compensation to be used is shown in Figure 3-2. Equation (41) will be used to calculate the required compensation; since in this case the feedback compensation transfer matrix, $\underset{\sim}{H}(\underset{\sim}{s})$, is the identity matrix.

$$\underset{\sim}{G}_c(s) = \{[\underset{\sim}{G}(s)^{-1} - \underset{\sim}{I}]\underset{\sim}{G}_p(s)\}^{-1} \qquad (41)$$

Since $\underset{\sim}{G}_p(s)$ is a 2x2 matrix in this example $\underset{\sim}{G}_c(s)$ and $\underset{\sim}{G}(s)$ will also be 2x2. By inspection (Equation (52)), $\underset{\sim}{G}(s)$ is non-singular and can be inverted.

The compensated system transfer matrix $\mathbf{G}(s)$ is given in equation (52). The inverse of $\mathbf{G}(s)$ is calculated by equation (62).

$$\mathbf{G}(s)^{-1} = \frac{\mathbf{COF}^T}{|\mathbf{G}(s)|} \tag{62}$$

Where $\mathbf{COF}^T$ is the transpose of the cofactor matrix of $\mathbf{G}(s)$, and $|\mathbf{G}(s)|$ is the determinant of $\mathbf{G}(s)$.

$$\mathbf{COF}^T \quad \begin{bmatrix} \dfrac{16}{s^2+4.8s+16} & 0 \\ 0 & \dfrac{100}{s^2+8s+100} \end{bmatrix} \tag{63}$$

$$|\mathbf{G}(s)| = \frac{(16)(100)}{(s^2+8s+100)(s^2+4.8s+16)} \tag{64}$$

$$\mathbf{G}(s)^{-1} = \begin{bmatrix} \dfrac{s^2+8s+100}{100} & 0 \\ 0 & \dfrac{s^2+4.8s+16}{16} \end{bmatrix} \tag{65}$$

$$[\mathbf{G}(s)^{-1}-\mathbf{I}] = \begin{bmatrix} \dfrac{s^2+8s}{100} & 0 \\ 0 & \dfrac{s^2+4.8s}{16} \end{bmatrix} \tag{66}$$

The plant transfer matrix, $\mathbf{G}_p(s)$, is given in equation (61) as:

$$\underset{\sim}{G}_p(\underset{\sim}{s}) = \begin{bmatrix} \dfrac{(s+2)}{(s+7)(s^2+14s+16)} & \dfrac{2}{(s+4.3)(s^2+14s+16)} \\ \\ \dfrac{4}{(s+7)(s^2+14s+16)} & \dfrac{(s+12)}{(s+4.3)(s^2+14s+16)} \end{bmatrix} \quad (61)$$

Multiplying equation (61) by equation (66):

$$[\underset{\sim}{G}(\underset{\sim}{s})^{-1}-\underset{\sim}{I}]\underset{\sim}{G}_p(\underset{\sim}{s}) = \begin{bmatrix} \dfrac{s(s+8)(s+2)}{100(s+7)(s^2+14s+16)} & \dfrac{2s(s+8)}{100(s+4.3)(s^2+14s+16)} \\ \\ \dfrac{4(s+4.8)}{16(s+7)(s^2+14s+16)} & \dfrac{s(s+4.8)(s+12)}{16(s+4.3)(s^2+14s+16)} \end{bmatrix} \quad (67)$$

If this matrix is non-singular it can be inverted.

$$|[\underset{\sim}{G}(\underset{\sim}{s})^{-1}-\underset{\sim}{I}]\underset{\sim}{G}_p(\underset{\sim}{s})| =$$

$$\frac{s^2(s+8)(s+2)(s+4.8)(s+12)}{(16)(100)(s+7)(s+4.3)(s^2+14s+16)^2} - \quad (68)$$

$$\frac{8s^2(s+4.8)(s+8)}{(16)(100)(s+7)(s+4.3)(s^2+14s+16)^2}$$

$$|[\underset{\sim}{G}(\underset{\sim}{s})^{-1} - \underset{\sim}{I}]\underset{\sim}{G}_p(\underset{\sim}{s})| =$$

$$\frac{s^2(s+4)(s+8)(s^2+14s+24-8)}{(16)(100)(s+7)(s+4.3)(s^2+14s+16)^2} = \quad (69)$$

$$\frac{s^2(s+4.8)(s+8)}{(16)(100)(s+7)(s+4.3)(s^2+14s+16)}$$

$$
\underset{\sim}{COF}^{T} = \begin{bmatrix} \dfrac{s(s+4.8)(s+12)}{16(s+4.3)(s^{2}+14s+16)} & \dfrac{-2s(s+8)}{100(s+4.3)(s^{2}+14s+16)} \\ \\ \dfrac{-4s(s+4.8)}{16(s+7)(s^{2}+14s+16)} & \dfrac{s(s+8)(s+2)}{100(s+7)(s^{2}+14s+16)} \end{bmatrix} \tag{70}
$$

$$
\underset{\sim}{G}_{c}(\underset{\sim}{s}) = \{ [\underset{\sim}{G}(\underset{\sim}{s})^{-1} - \underset{\sim}{I}]\underset{\sim}{G}_{p}(\underset{\sim}{s})\}^{-1} =
$$

$$
\begin{bmatrix} \dfrac{100(s+7)(s+12)}{s(s+8)} & \dfrac{-32(s+7)}{s(s+4.8)} \\ \\ \dfrac{-400(s+4.3)}{s(s+8)} & \dfrac{16(s+2)(s+4.3)}{s(s+4.8)} \end{bmatrix} \tag{71}
$$

One of the possible realizations of this compensation is shown in Figure 3-3. To obtain a confirmation of the design, the system was simulated with and without compensation using International Business Machines' Digital Simulation Language, DSL (13). Figures 3-4 through 3-9 show the results of that simulation, which indicate that the compensators used do in fact achieve the stated objectives of the design. In each plot, trace number 1 is the desired response, Z, and trace number 2 the system response, Y. In Figures 3-4B, 3-7B, 3-8B, and 3-9B the two traces are superimposed. In Figures 3-5A and 3-6A the desired responses are zero. The reader should note the scale factors present on some of these plots.

2. Parameter Optimization Solution

The compensated system and the desired response curves were then simulated and solutions were obtained using the

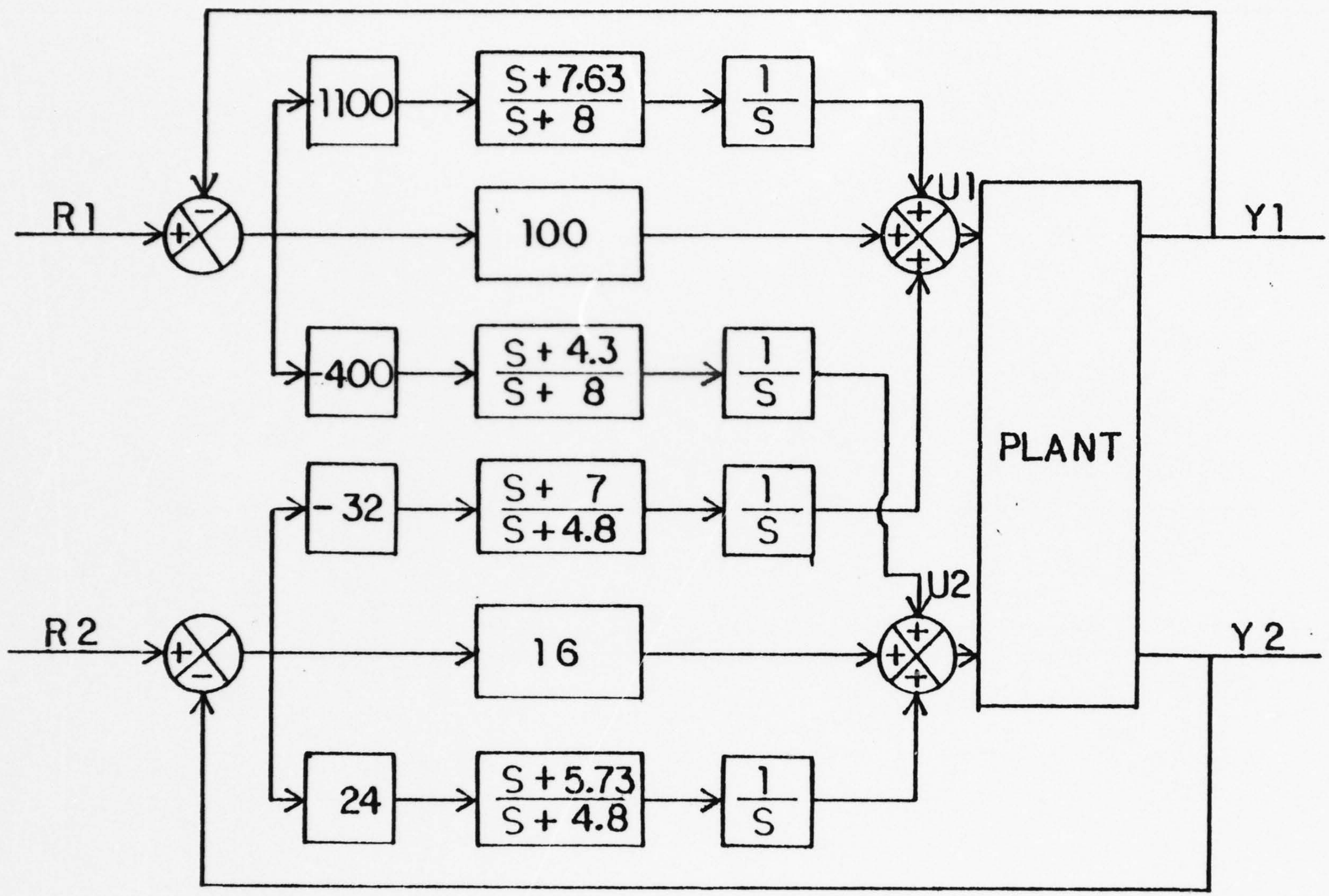

Figure 3-3 Compensated Multivariable Plant

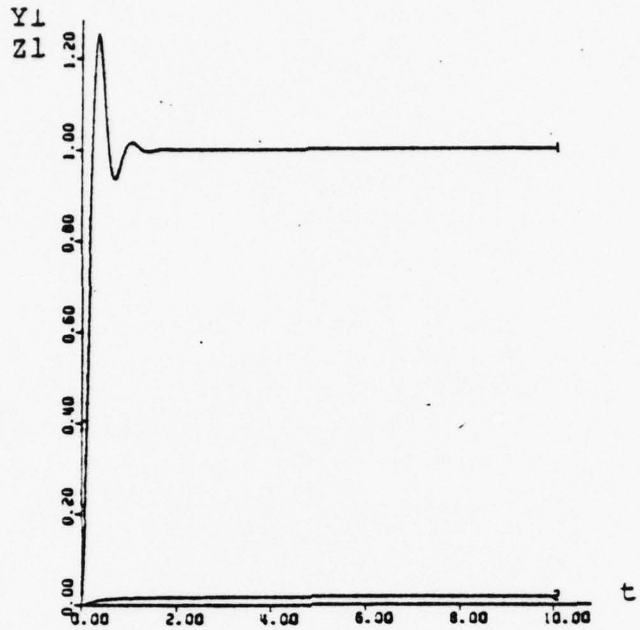

Figure 3-4A

Y1 and Z1 vs Time with R1 = Unit Step, R2 = 0.0

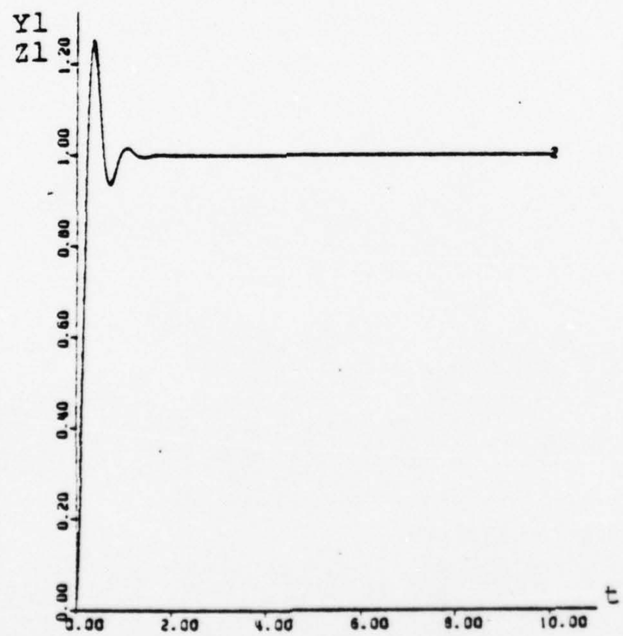

Figure 3-4B

Y1 and Z1 vs Time with R1 = Unit Step, R2 = 0.0
For Compensated System

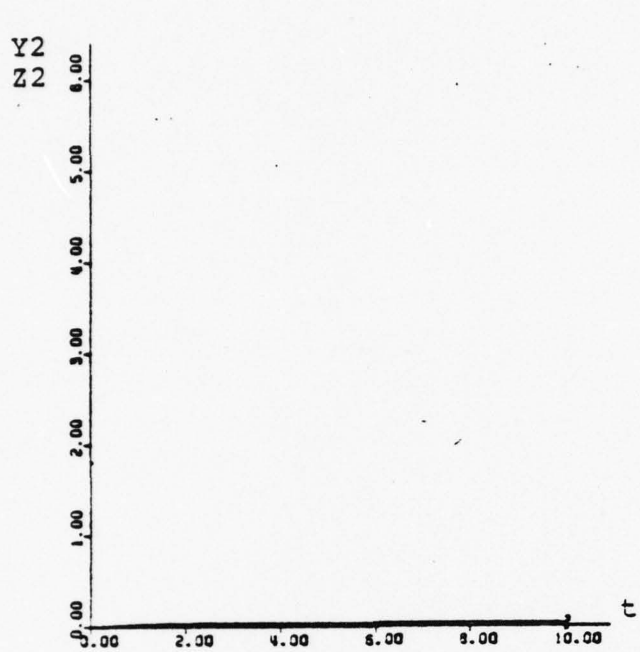

Figure 3-5A

Y2 and Z2 vs Time with R1 = Unit Step, R2 = 0.0

($Y2_{ss} = 3.57 \times 10^{-2}$)

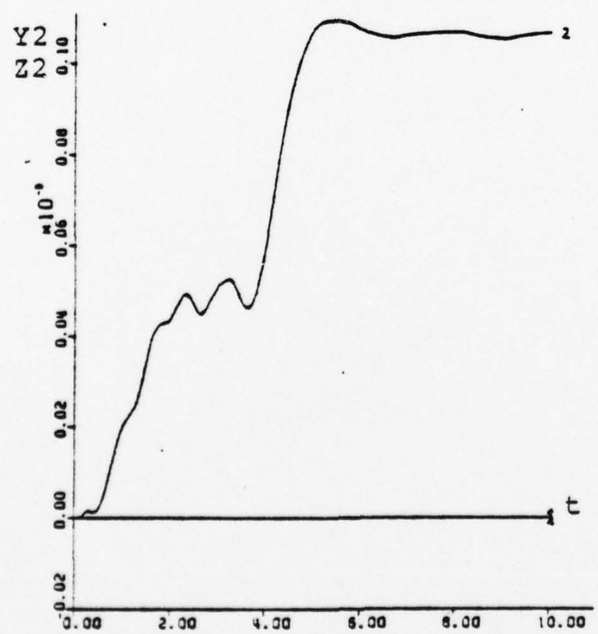

Figure 3-5B

Y2 and Z2 vs Time with R1 = Unit Step, R2 = 0.0
For Compensated System

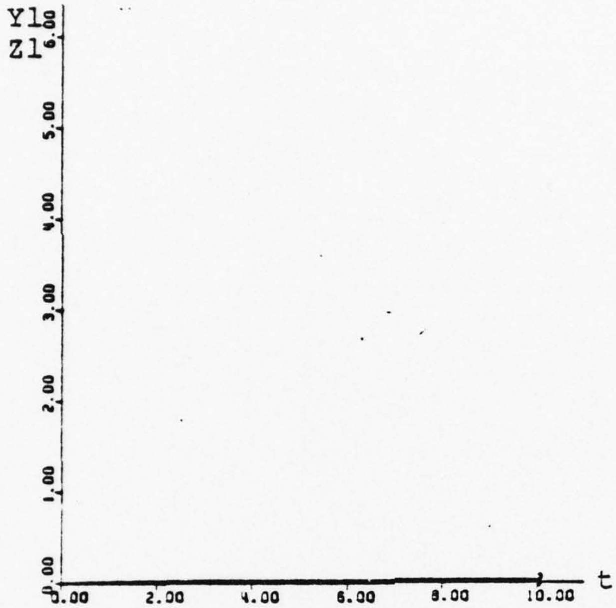

Figure 3-6A

Y1 and Z1 vs Time with R1 = 0.0, R2 = Unit Step

($Y1_{ss} = 2.91 \times 10^{-2}$)

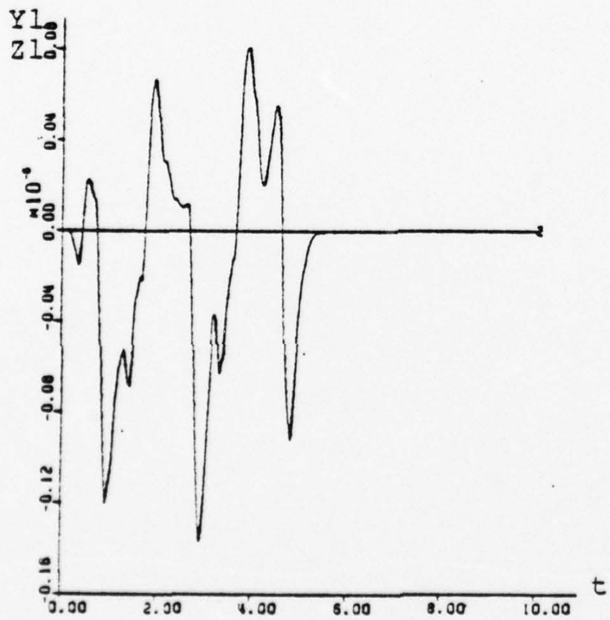

Figure 3-6B

Y1 and Z1 vs Time with R1 = 0.0, R2 = Unit Step
For Compensated System

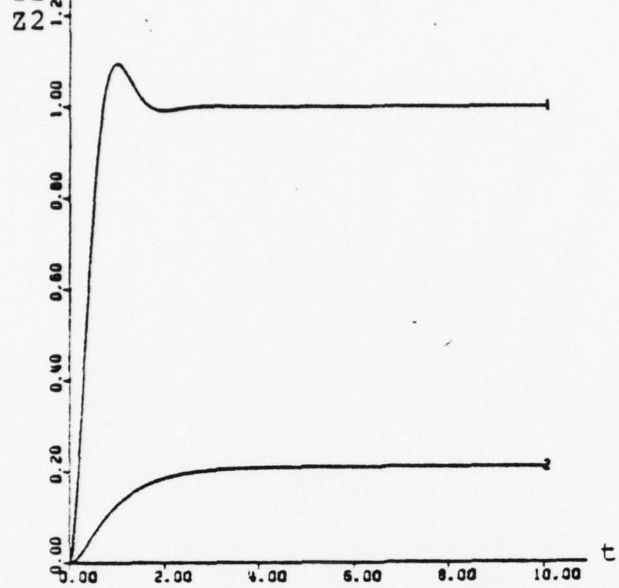

Figure 3-7A

Y2 and Z2 vs Time with R1 = 0.0, R2 = Unit Step

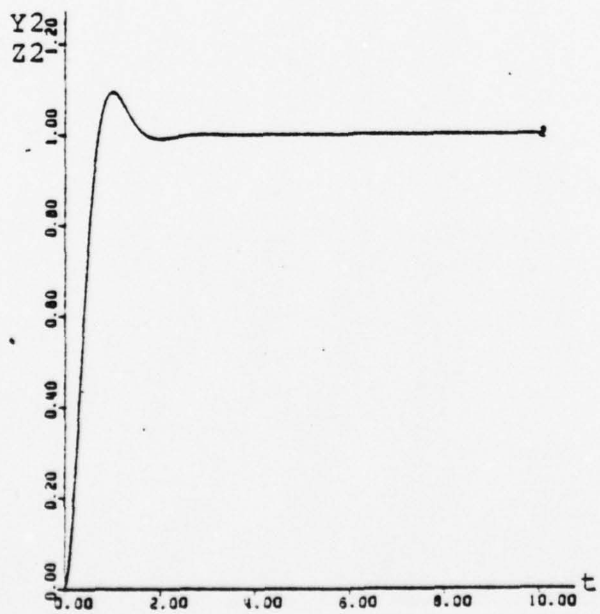

Figure 3-7B

Y2 and Z2 vs Time with R1 = 0.0, R2 = Unit Step
For Compensated System

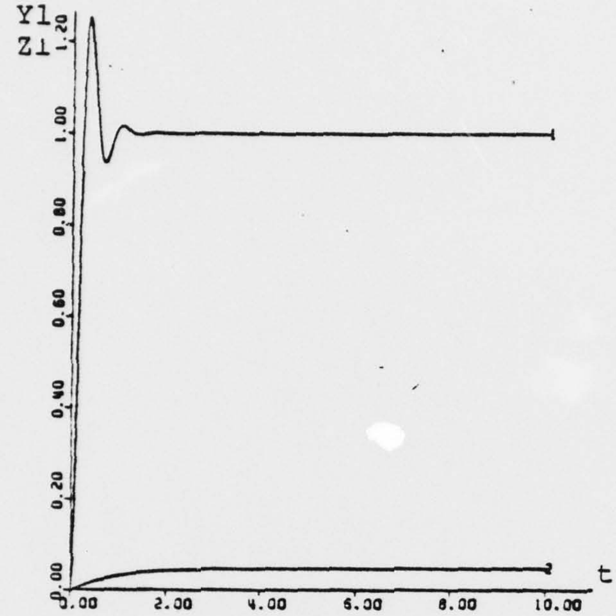

Figure 3-8A

Y1 and Z1 vs Time with R1 = Unit Step, R2 = Unit Step

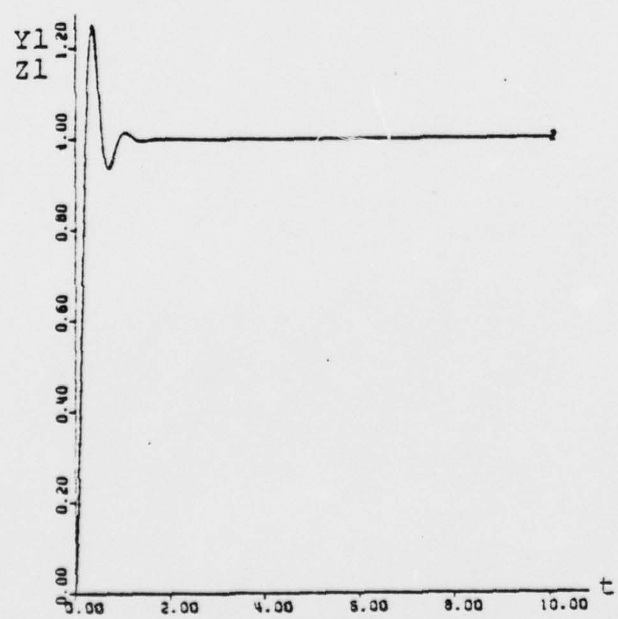

Figure 3-8B

Y1 and Z1 vs Time with R1 = Unit Step, R2 = Unit Step
For Compensated System

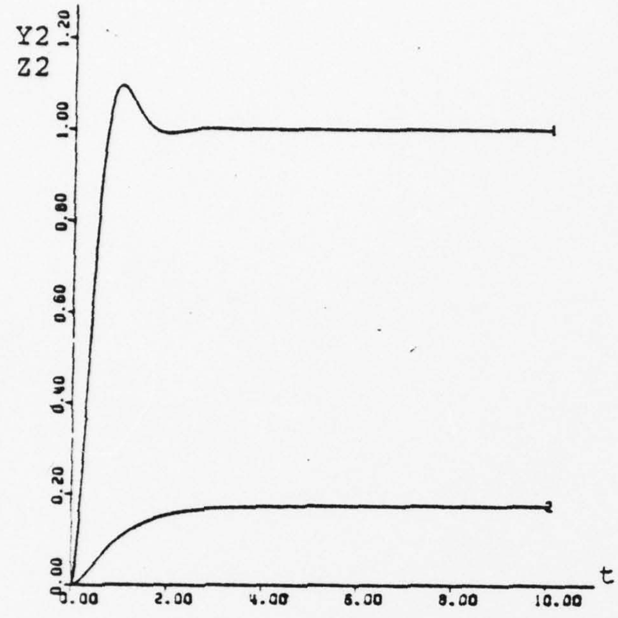

Figure 3-9A

Y2 and Z2 vs Time with R1 = Unit Step, R2 = Unit Step

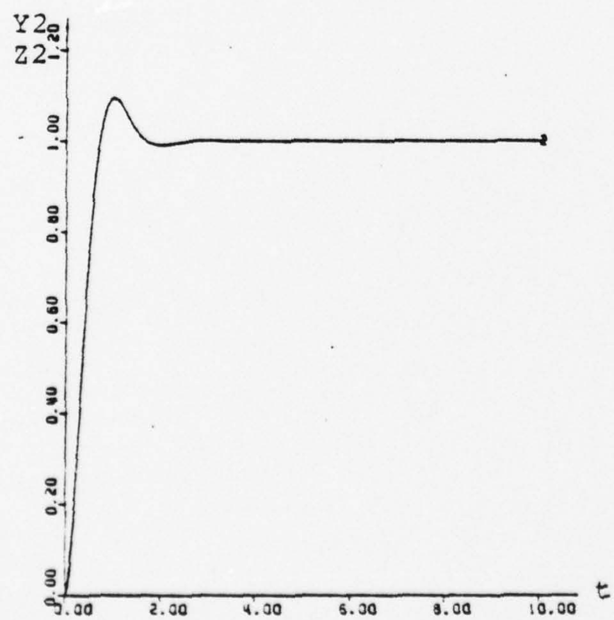

Figure 3-9B

Y2 and Z2 vs Time with R1 = Unit Step, R2 = Unit Step
For Compensated System

program discussed in this section. The time response results are shown in Figure 3-10. There is a slight difference between the desired response curve and the output of the simulated system.

Next, two parameter optimization was performed. The proportional gain constant of each compensator was selected as the adjustable parameters. Each of the four performance measures given by equations (47) through (51) were used in successive optimizations. All other aspects of the program were unchanged during this optimization. Equal weighting was applied to each channel. The gains 100 and 16 were chosen because their variation should have similar impact on the two respective outputs. The upper limits were 105 and 21 and the lower limits were 95 and 11 respectively. The integration step size was .02 and the final time was 5 seconds. Optimization was begun with the two gains set at 105 and 11 respectively, and R1 = R2 = Unit Step functions. The results are given in Table 3-1.

The system was simulated using these values for the two free parameters. The simulation results are shown in Figures 3-11 through 3-14. This simulation indicates the time multiplied integral square-error performance measure produced the best results.

Performance Measure	Value of Minimum Performance Measure	G_{100}	G_{16}
Integral Square-Error	1.2026097×10^{-3}	104.9999	20.99998
Time Multiplied Integral Square-Error	1.641641×10^{-5}	102.2554	15.27091
Integral Absolute Error	1.353197×10^{-3}	104.9862	14.82275
Time Multiplied Integral Absolute Error	9.994698×10^{-4}	103.2728	14.81198

Table 3-1

RESULTS OF OPTIMIZATION

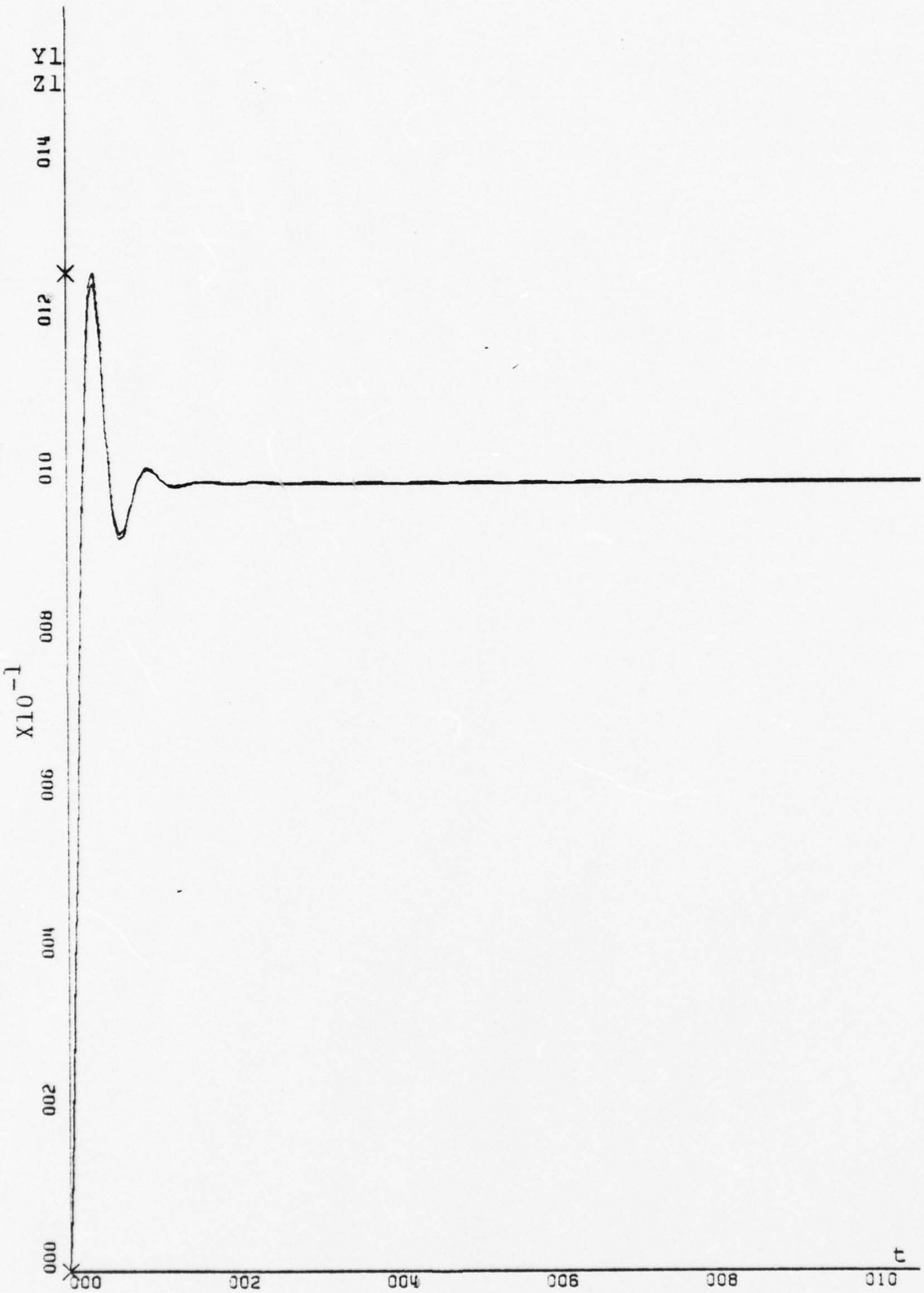

Figure 3-10A

Y1 and Reference 1 vs Time Simulated with Thesis Program Parameters Set at Analytical Optimum

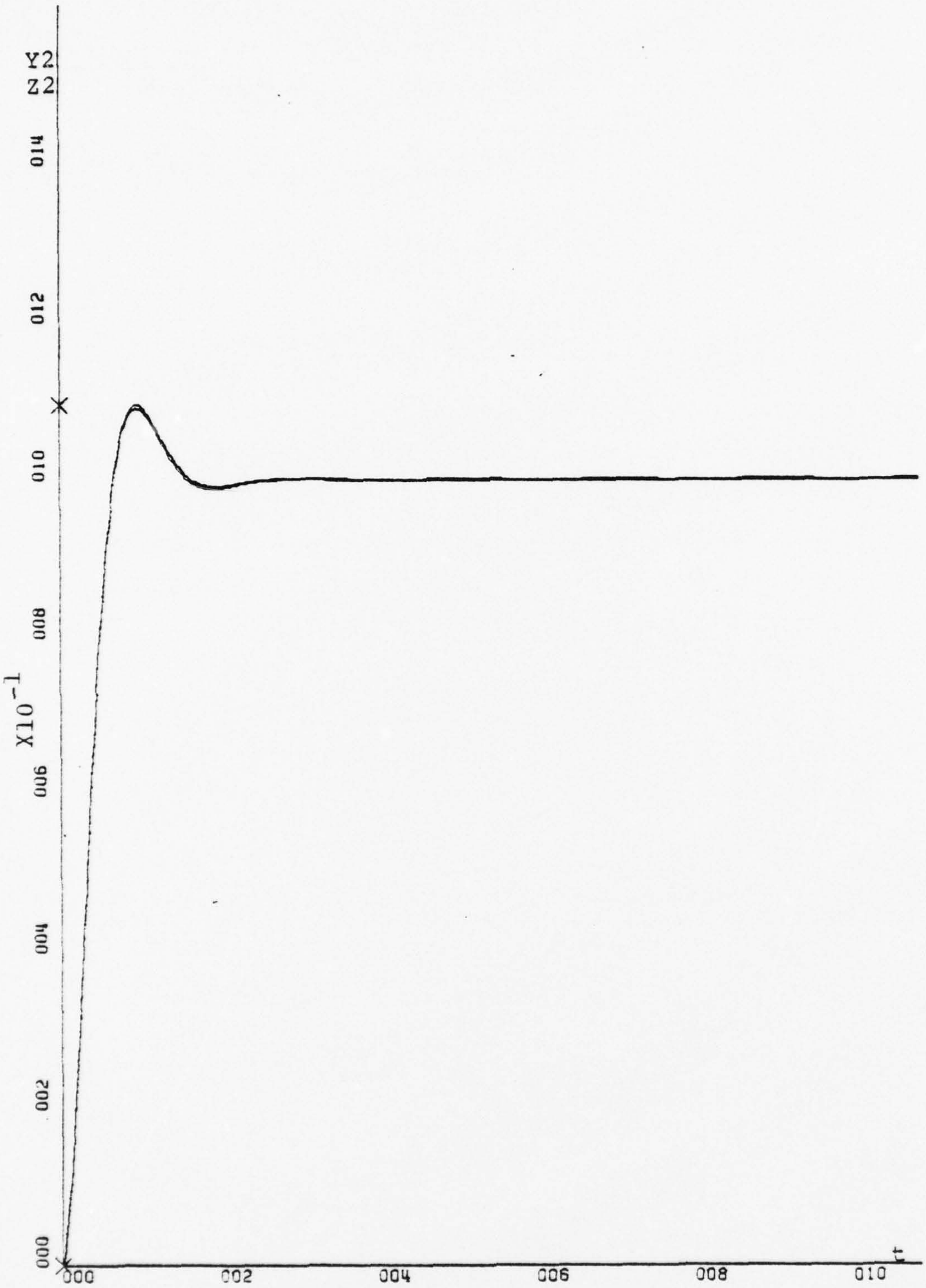

Figure 3-10B

Y2 and Reference 2 vs Time Simulated with Thesis Program Parameters Set at Analytical Optimum

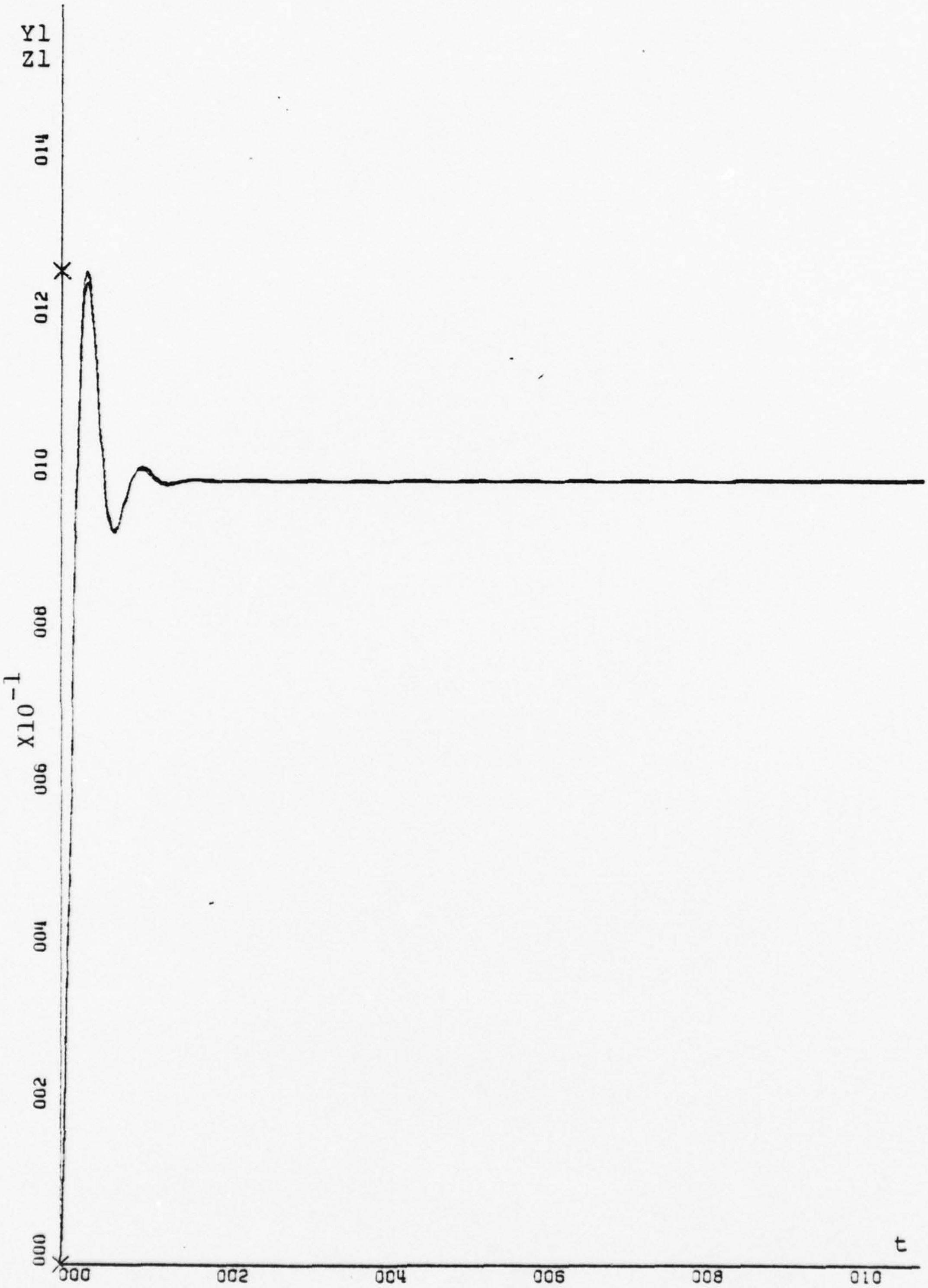

Figure 3-11A

Y1 and Reference 1 vs Time After Optimization With Integral-Square Error Performance Measure

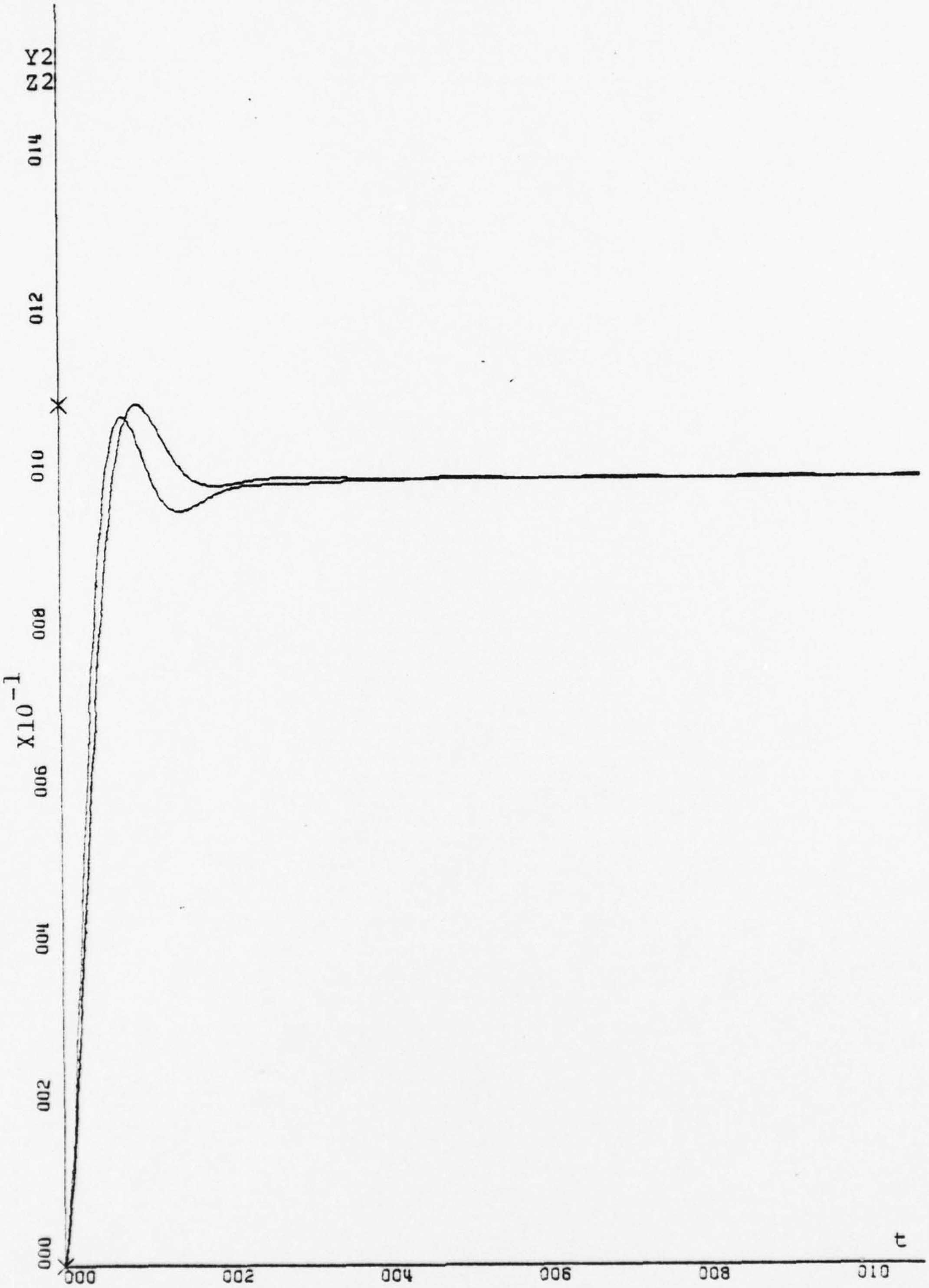

Figure 3-11B

Y2 and Reference 2 vs Time After Optimization With Integral Square-Error Performance Measure (Curve With Greatest Overshoot is ZZ)

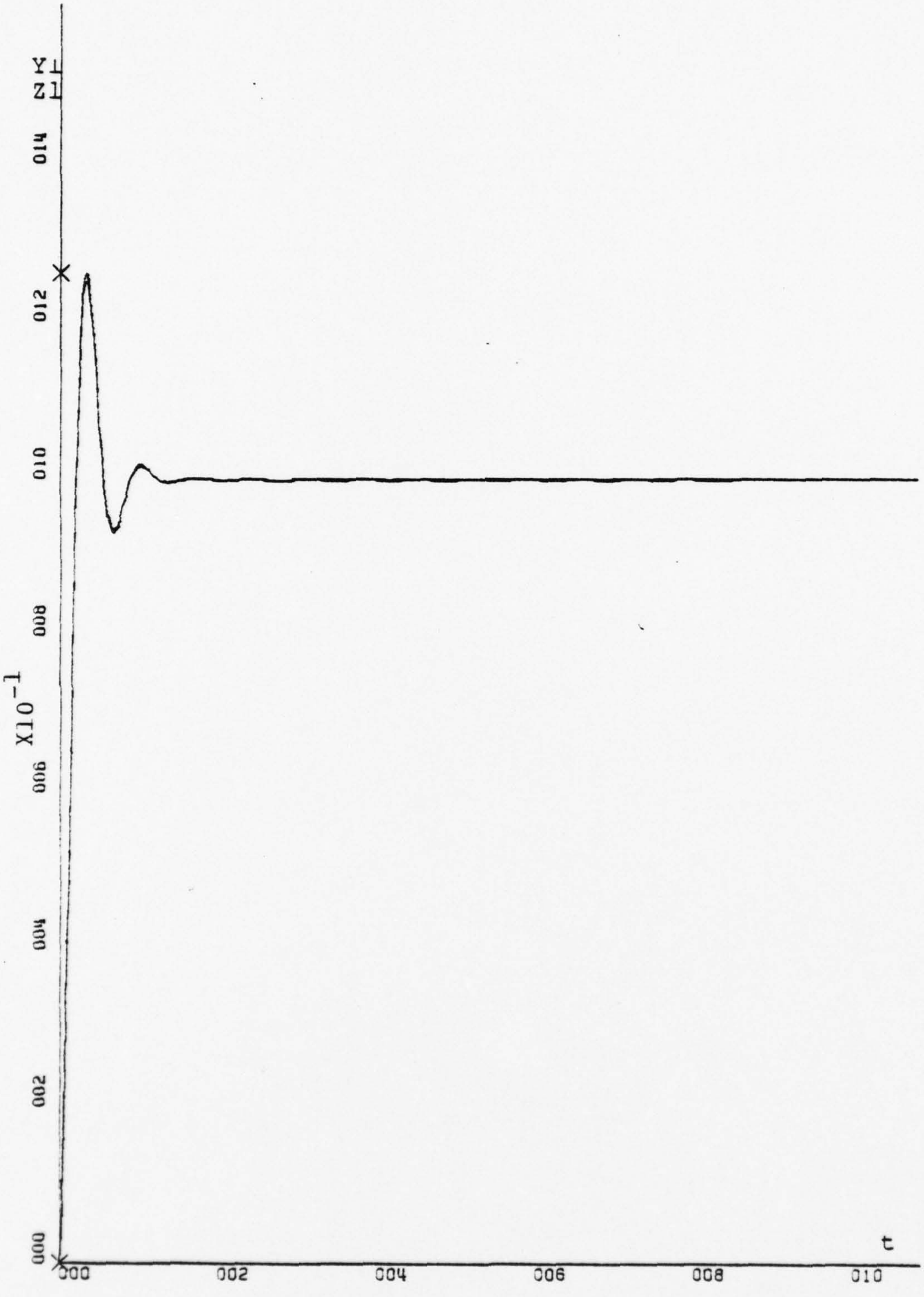

Figure 3-12A

Y1 and Reference 1 vs Time After Optimization With
Time Multiplied Integral-Square Error Performance Measure

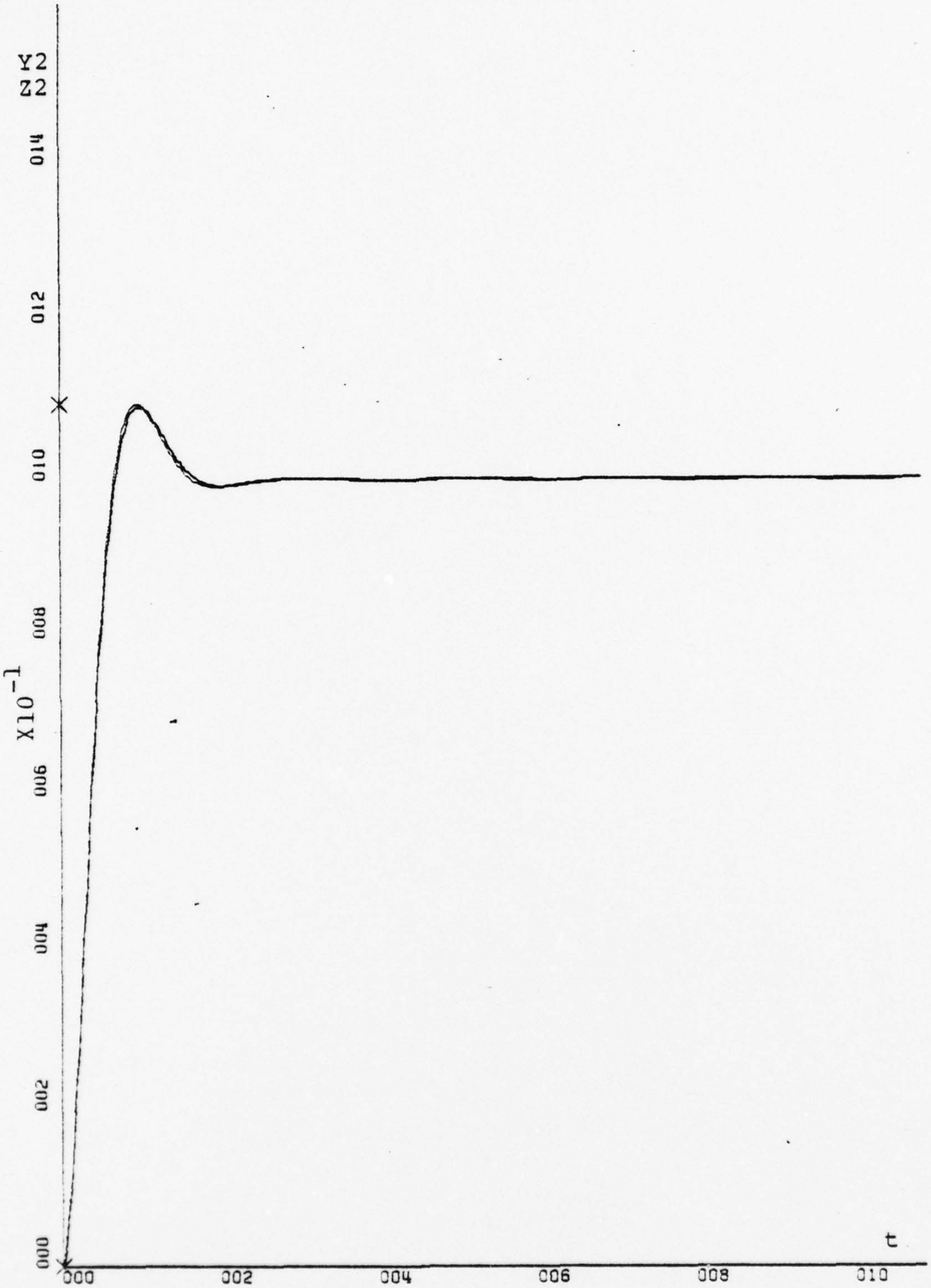

Figure 3-12B

Y2 and Reference 2 vs Time After Optimization With Time Multiplied Integral Square-Error Performance Measure

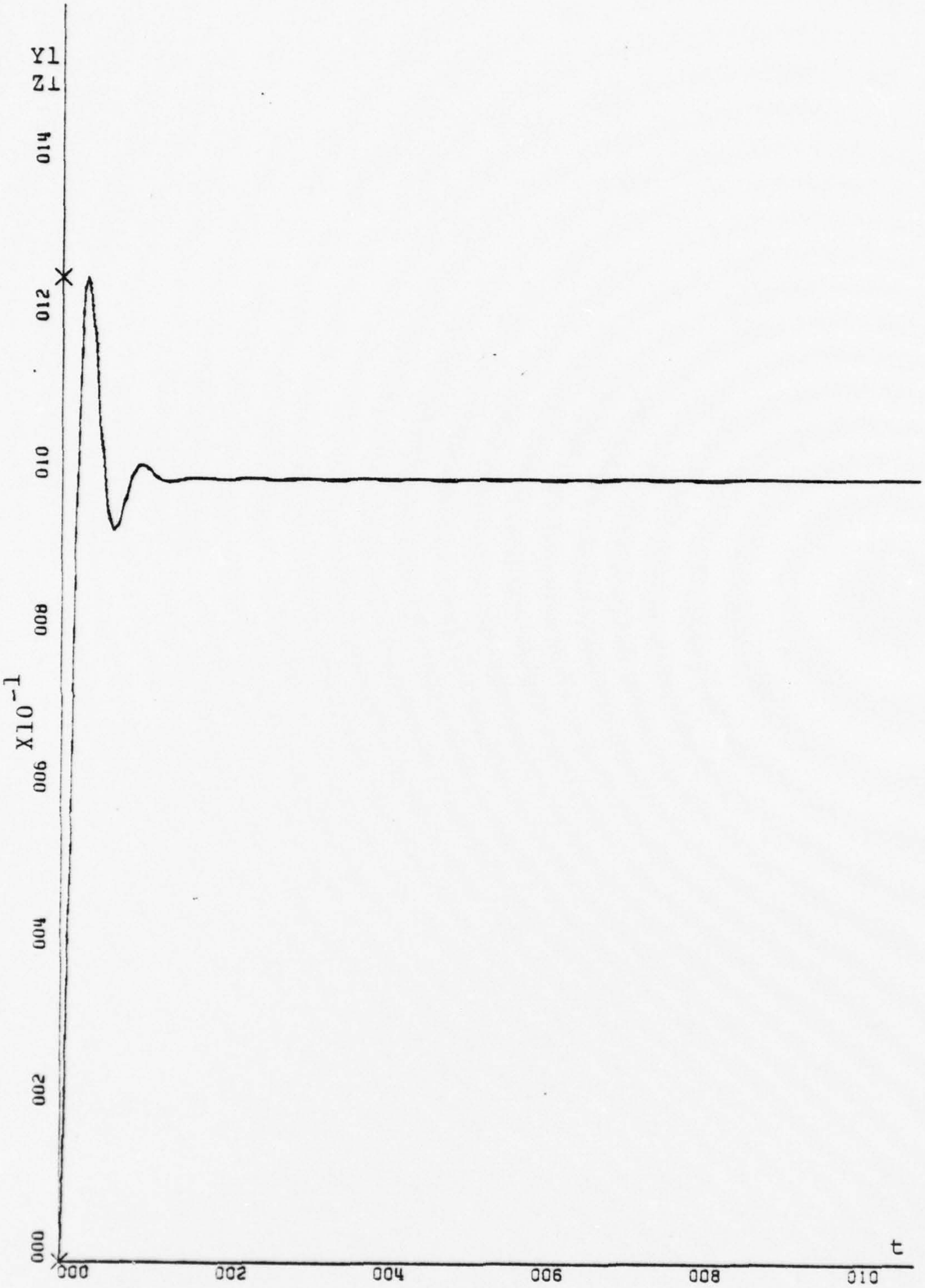

Figure 3-13A

Y1 and Reference 1 vs Time After Optimization With Integral Absolute Error Performance Measure

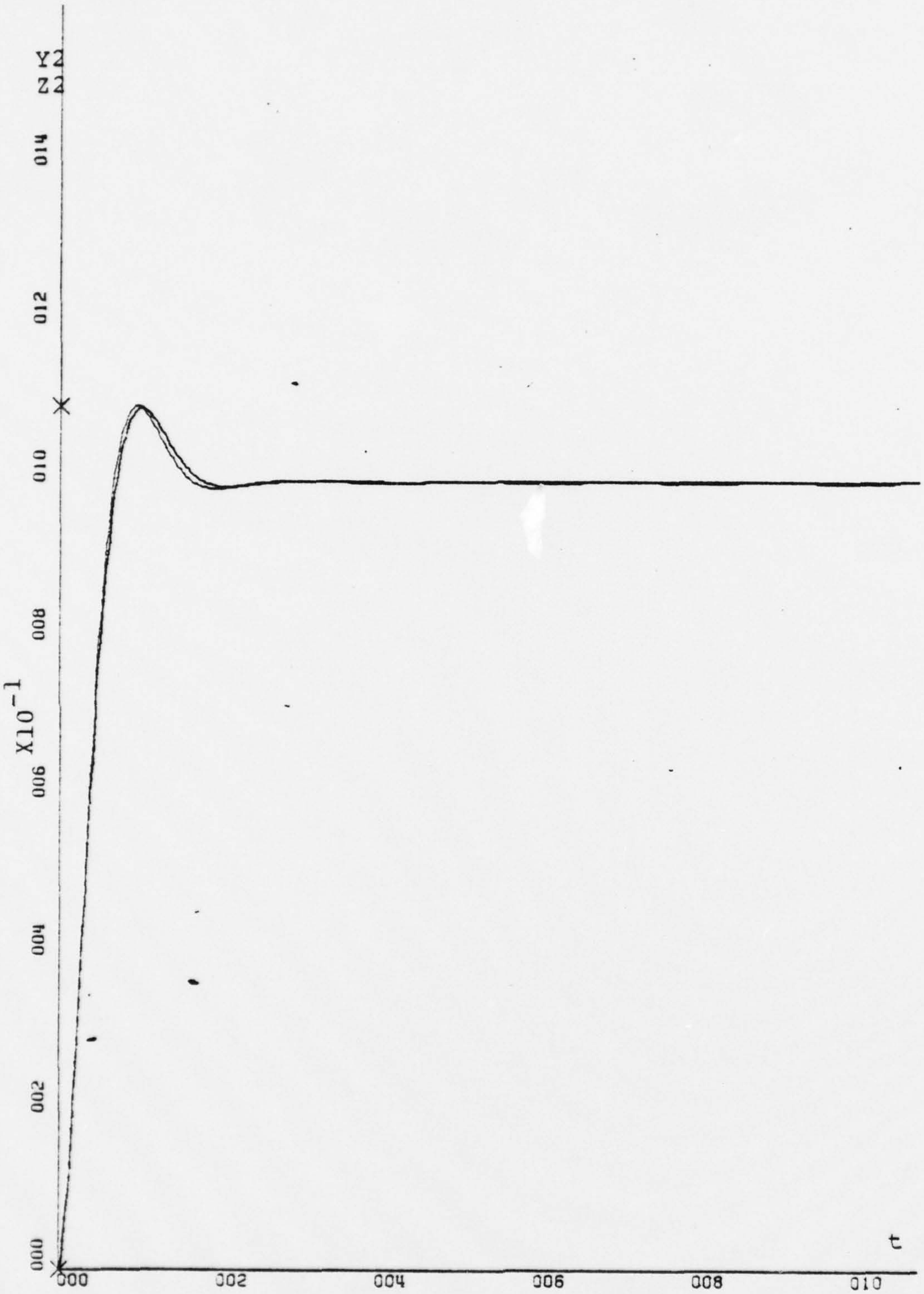

Figure 3-13B

Y2 and Reference 2 vs Time After Optimization With Integral Absolute Error Performance Measure

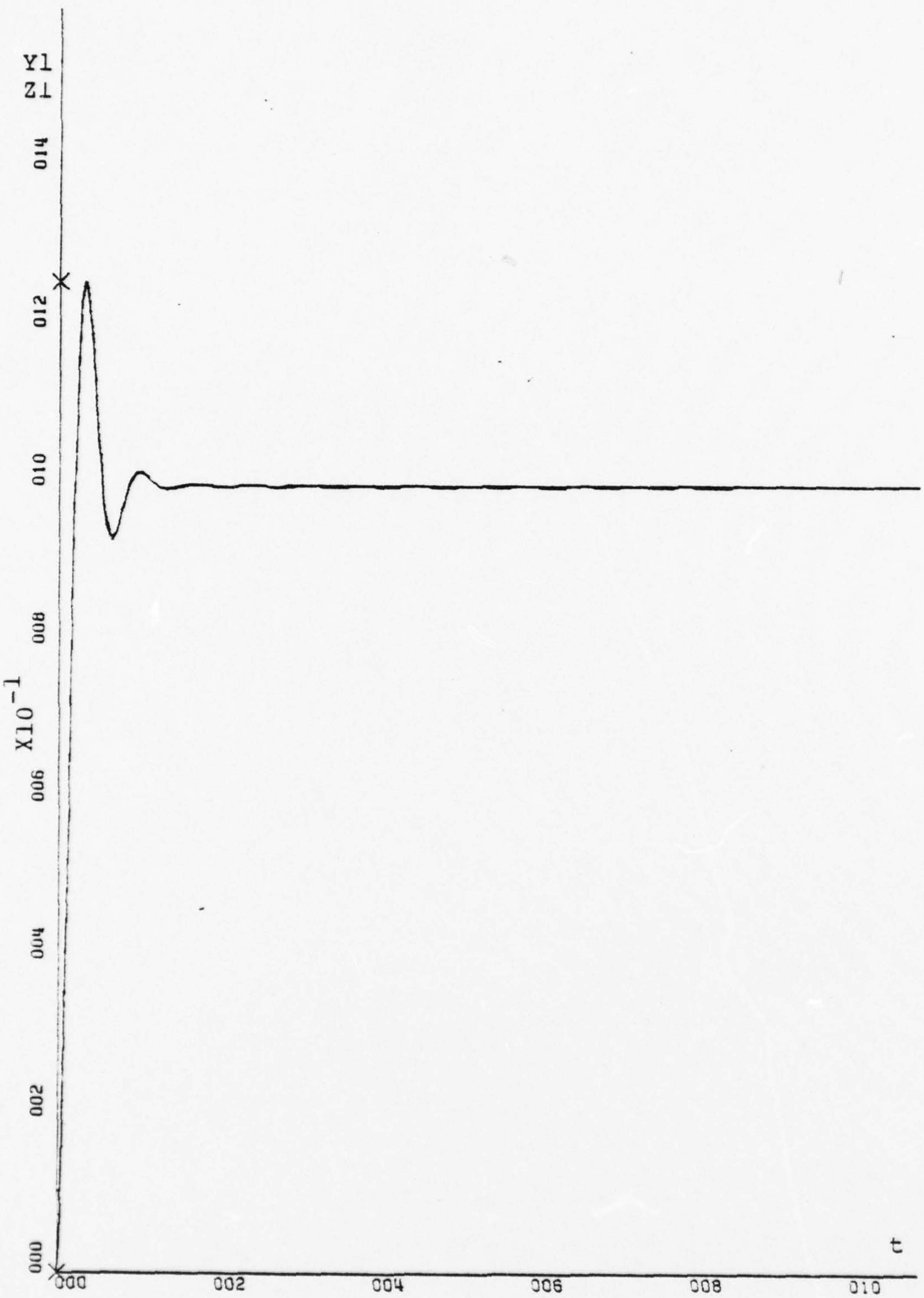

Figure 3-14A

Y1 and Reference 1 vs Time After Optimization With Time Multiplied Integral Absolute Error Performance Measure

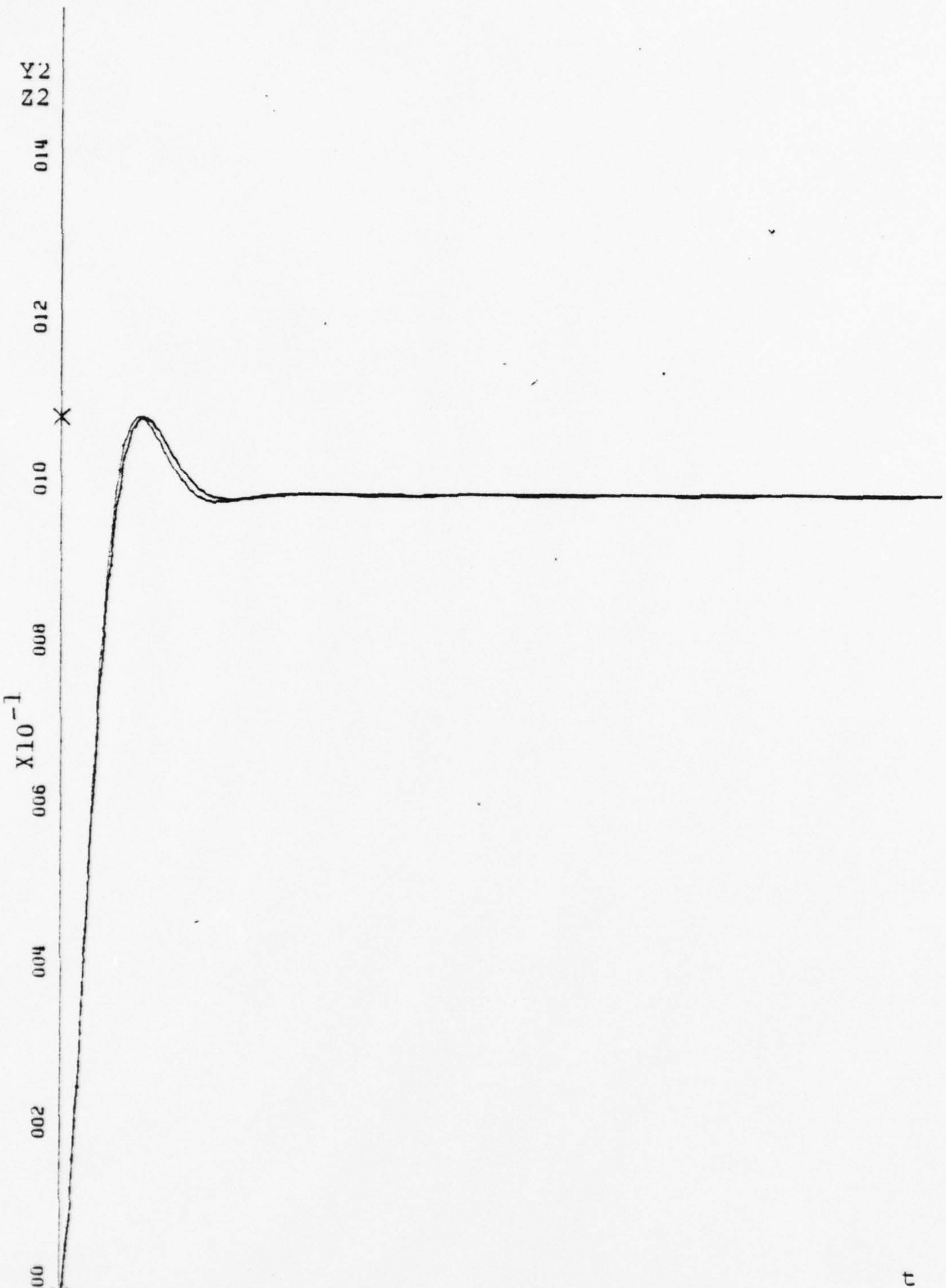

Figure 3-14B

Y2 and Reference 2 vs Time After Optimization With
Time Multiplied Integral Absolute Error Performance Measure

CONCLUSIONS

From the results of the two-input two-output example system, the following conclusions can be stated:

1. The technique of cost function minimization in the time domain can be used effectively for compensator parameter optimization in multiple input multiple output control systems.

2. Cost function minimization is not a substitute for control system design. The user must be able to select the proper type of compensator which is capable of achieving the desired results, before initiating the optimization of the free parameters.

3. The four performance measures evaluated produced slightly different solutions to the same problem, with the time multiplied integral square-error yielding the best results in the example given.

APPENDIX A

```
C*************************************************************************C
C                                                                         C
C                   COMPENSATOR OPTIMIZATION IN                           C
C                                                                         C
C                 MULTIVARIABLE CONTROL SYSTEMS                           C
C                                                                         C
C                              BY                                         C
C                                                                         C
C                        JOHN T. MOWREY                                   C
C                                                                         C
C                          MAJOR USMC                                     C
C                                                                         C
C                        DECEMBER 1978                                    C
C                                                                         C
C THIS PROGRAM WAS DEVELOPED AS A GENERAL PURPOSE DESIGN                  C
C AID TO BE USED TO OPTIMIZE FREE PARAMETERS IN THE COMP-                 C
C ENSATORS OF A MULTIVARIABLE CONTROL SYSTEM. IT CAN ALSO                 C
C BE USED FOR THE OPTIMIZATION OF FREE PARAMETERS IN A                    C
C SINGLE INPUT SINGLE OUTPUT SYSTEM OR TO SIMULATE EITHER                 C
C TYPE OF SYSTEM. OPTIMIZATION IS ACCOMPLISHED BY MINIMIZ-                C
C ING THE ERROR BETWEEN A USER SPECIFIED REFERENCE CURVE                  C
C AND THE OUTPUT OF THE SIMULATED COMPENSATED SYSTEM.                     C
C                                                                         C
C*************************************************************************C
C                                                                         C
C                          DATA CARDS                                     C
C                          **********                                     C
C                                                                         C
C CARD 1  NRUNS,NOPT,NGRAPH,NREF,IIN,NOUT,IPRINT,IFREQ                    C
C                                                                         C
C         FORMAT(8I5)                                                     C
C                                                                         C
C         NRUNS-FLAG FOR NUMBER OF RUNS. NRUNS MUST EQUAL                 C
C         1 WHEN OPTIMIZING.                                              C
C                                                                         C
C         NOPT-FLAG FOR OPTIMIZATION/SIMULATION                           C
C            1-OPTIMIZATION                                               C
C            2-SIMULATION ONLY                                            C
C            (IF NOPT=1 THE PLANT WILL BE SIMULATED AFTER                 C
C            OPTIMIZATION AND THE REQUESTED OUTPUT PRODUCED               C
C                                                                         C
C         NGRAPH-FLAG FOR TYPE OF GRAPHICAL OUTPUT DESIRED                C
C            1-PRODUCES PRINTER PLOTS                                     C
C            2-PRODUCES VERSATEC PLOTS                                    C
C            3-NO GRAPHS PRODUCED                                         C
C                                                                         C
C         NREF-FLAG FOR REFERENCE CURVE                                   C
C            1-PRODUCES CURVE                                             C
C            2-NO REFERENCE CURVE IS PRODUCED                             C
C                                                                         C
C         IIN-THE NUMBER OF INPUTS (UP TO 25)                             C
C                                                                         C
C         NOUT-THE NUMBER OF OUTPUTS (UP TO 3)                            C
C                                                                         C
C         IPRINT-FLAG FOR TABULATED DATA FOR REFERENCE                    C
C         CURVE AND SYSTEM RESPONSE.                                      C
C            1-PRODUCES PRINTED OUTPUT                                    C
C            2-NO OUTPUT DATA PRINTED                                     C
C                                                                         C
C         IFREQ-FREQUENCY OF PRINTED OUTPUT. IF IFREQ=10                  C
C            EVERY 10TH DATA POINT WILL BE PRINTED.                       C
C                                                                         C
C                                                                         C
C CARD 2  T47,DT,TF                                                       C
C                                                                         C
C         FORMAT(3F10,5)                                                  C
C                                                                         C
C         T47-INITIAL TIME                                                C
C                                                                         C
C         DT-INTEGRATION STEP SIZE                                        C
C                                                                         C
C         TF-FINAL TIME                                                   C
```

```
CARD 3   NV,NAV,NTA,NPR,IP,IPM

         FORMAT(6I5)

         THIS CARD MUST BE OMITTED IF NOPT=2

         NV-THE NUMBER OF FREE VARIABLES TO BE SET BY
           OPTIMIZATION.

         NAV-THE NUMBER OF AUXILLIARY VARIABLES

         NTA-THE NUMBER OF TRIALS ALLOWED BOXPLX

         NPR-FREQUENCY OF OUTPUT FROM BOXPLX FOR
           DIAGNOSTIC PURPOSES

         IP-FLAG FOR INTEGER/REAL*8 OPTIMIZATION.

                           *       *
                           *  NOTE *
                           *       *

SEE BOXPLX FOR ADDITIONAL INFORMATION ON THE 5 VARIABLES
LISTED ABOVE. BOXPLX WAS CONVERTED TO DOUBLE PRECISION
FOR USE IN THIS PROGRAM: HOWEVER THE DOCUMENTATION WAS
NOT CHANGED

         IPM-FLAG FOR PERFORMANCE MEASURE.

           THE PERFORMANCE MEASURES USED ARE PROPORTIONAL
           TO THOSE INDICATED BELOW.

           1-INTEGRAL SQUARE-ERROR (WEIGHTED)
           2-TIME MULTIPLIED INTEGRAL SQUARE-ERROR
             (WEIGHTED)
           3-INTEGRAL ABSOLUTE ERROR (WEIGHTED)
           4-TIME MULTIPLIED INTEGRAL ABSOLUTE ERROR
             (WEIGHTED)

CARDS 4-7 XS(I)

         FORMAT(8F10.5)

         ALL OF THESE CARDS MUST BE OMITTED IF NOPT=2
         XS(I) ARE THE STARTING VALUES OF THE FREE PARA-
         METERS. THE NUMBER OF CARDS REQUIRED IS DEPEN-
         DENT ON NV. ONE VALUE MUST BE READ IN FOR EACH
         FREE PARAMETER. IE IF NV=8 ONLY CARD 4 IS RE-
         QUIRED AND 8 PARAMETER VALUES SHOULD BE ON IT
         IN 8F10.5 FORMAT.

CARDS 8-11 XU(I)

         FORMAT (8F10.5)

         THESE CARDS ARE THE SAME AS CARDS 4-7 EXCEPT
         XU(I) ARE THE UPPER LIMITS ON THE FREE PARA-
         METERS. OMIT IF NOPT=2.

CARDS 12-15 XL(I)

         FORMAT(8F10.5)

         THESE CARDS ARE THE SAME AS CARDS 4-7 EXCEPT
         XL(I) ARE THE LOWER LIMITS ON THE FREE PARA-
         METERS. OMIT IF NOPT =2.
```

```
C CARDS 16,18, AND 20 IOUT,IWT,ITAB                                    C
C                                                                      C
C         FORMAT(3F10.5)                                               C
C                                                                      C
C         IOUT-THE NUMBER OF THE BLOCK FROM WHICH THE OUT-             C
C            PUT IS TAKEN. IF 2 OR MORE BLOCKS FEED AN OUT-            C
C            PUT NODE,A TYPE 1 BLOCK WITH G=1 MUST BE                  C
C            PLACED BETWEEN THAT NODE AND THE OUTPUT.                  C
C                                                                      C
C         FOR A 3 OUTPUT SYSTEM ALL 3 CARDS ARE REQUIRED.              C
C         FOR A 2 OUTPUT SYSTEM 16 AND 18 ARE REQUIRED.                C
C                                                                      C
C         IWT-AN INTEGER WEIGHTING FUNCTION WHICH WILL                 C
C            BE CONVERTED TO REAL*8.AND WHICH ALLOWS THE               C
C            USER TO PENALIZE THE ERROR BETWEEN THE SYSTEM             C
C            OUTPUT AND THE REFERENCE RESPONSE MORE HEAVILY            C
C            AT ONE OUTPUT THAN ANOTHER. INTEGERS BETWEEN 1            C
C            AND 10 ARE RECOMMENDED. WEIGHTING SHOULD BE               C
C            REDUCED IF OVERFLOW IS ENCOUNTERED.                       C
C                                                                      C
C         ITAB-A FLAG FOR TABLE LOOK UP OF THE REFERENCE               C
C            CURVES.                                                   C
C               1-PROGRAM WILL READ TABULATED DATA FOR THE             C
C                  REFERENCE CURVES.                                   C
C               2-PROGRAM WILL CALCULATE A SECOND ORDER RES-           C
C                  PONSE CURVE.                                        C
C                                                                      C
C CARDS 17,19, AND 21-BETA,DELTA,WN,AMP,DELAY,INPUT                    C
C                                                                      C
C         FORMAT(5F10.5,A4)                                            C
C                                                                      C
C         THESE CARDS INPUT THE DATA REQUIRED TO COMPUTE A             C
C         SECOND ORDER RESPONSE CURVE. IF THE USER DESIRES             C
C         THE TABLE LOOK UP FEATURE THESE CARDS ARE RE-                C
C         PLACED BY TABULATED DATA. IF A TABULATED REF-                C
C         ERENCE CURVE IS DESIRED FOR OUTPUT NO. 1, CARD               C
C         17 IS REPLACED BY TABULATED DATA IN 8F10.5 FORMAT            C
C         THE NUMBER OF DATA POINTS REQUIRED IS TF/DT (TF              C
C         AND DT FROM CARD 2).                                         C
C                                                                      C
C           C(I)               BETA(I)                                 C
C          ------ = -------------------------------------              C
C           R(I)     (S**2+2*DELTA(I)*WN(I)*S+WN(I)**2)                C
C                                                                      C
C           C(I) IS THE DESIRED REFERENCE RESPONSE ASSOC-              C
C              IATED WITH OUTPUT IOUT(I).                              C
C           R(I) IS THE FORCING FUNCTION SPECIFIED BY:                 C
C              AMP(I)-THE AMPLITUDE OF THE FORCING FUNCTION            C
C              DELAY(I)-DELAY AFTER T47 BEFORE THE FORCING             C
C                 FUNCTION IS APPLIED(TIME DOMAIN).                    C
C              INPUT(I)-THE TYPE OF FORCING FUNCTION:STEP,             C
C                 RAMP, OR PARA.                                       C
C           BETA(I)-THE GAIN OF THE TRANSFER FUNCTION.                 C
C           DELTA(I)-THE DAMPING FACTOR.                               C
C           WN(I)-THE NATURAL FREQUENCY.                               C
C                                                                      C
C CARD 22 N                                                            C
C                                                                      C
C         FORMAT(I2)                                                   C
C                                                                      C
C         THE NUMBER OF BLOCKS IN THE SYSTEM (UP TO 25).               C
C                                                                      C
C CARDS 23-47 IC(I),ID(I),IE(I),IV(I),G(I),P(I),Z(I)                   C
C                                                                      C
C                                                                      C
C         FORMAT(4I2,2X,3F10.5)                                        C
C                                                                      C
C         THE NUMBER OF CARDS REQUIRED IS DETERMINED BY N.             C
C                                                                      C
C         IC(I)-THE NUMBER OF THE BLOCK(1-25).                         C
C                                                                      C
```

```
C     ID(I)-THE TYPE OF BLOCK. SIX TYPES ARE AVAILABLE   C
C                                                        C
C                   IE(I)     ----------  IV(I)          C
C       ID(I)=1     ------|       G      |------         C
C                             ----------                 C
C                                                        C
C                             ----------                 C
C                   IE(I)    |    G      |  IV(I)        C
C       ID(I)=2     ------|      ---     |------         C
C                            |   S+P     |               C
C                             ----------                 C
C                                                        C
C                             ----------                 C
C                   IE(I)    |    S+Z    |  IV(I)        C
C       ID(I)=3     ------|   G  ---     |------         C
C                            |    S+P    |               C
C                             ----------                 C
C                                                        C
C                         ------------------------------ C
C               IE(I)    |               G              |IV(I) C
C       ID(I)=4 ------|    -------------------------    |------C
C                        | S**2+2*DELTA*WN*S+WN**2      | C
C                         ------------------------------ C
C                                                        C
C                         ----------------------         C
C                        |        UL    /----   |        C
C                        |             /|       |        C
C                   IE(I)|            /-|G      | IV(I)  C
C       ID(I)=5     ------|          /          |-----   C
C                        |          /           |        C
C                        |   ----/   LL         |        C
C                         ----------------------         C
C                                                        C
C                IV(I)=G*IE(I) FOR G*IE(I) LE UL         C
C                              AND G*IE(I) GE LL         C
C                IV(I)=UL      FOR G*IE(I) GT UL         C
C                IV(I)=LL      FOR G*IE(I) LT UL         C
C                                                        C
C                         ------------------             C
C                        |                  /|           C
C                        |                 / |           C
C                        |             B  /  |           C
C                        |              |    |           C
C                   IE(I)|              |    | IV(I)     C
C       ID(I)=6     ------|    ----------    |------     C
C                        |    |              |           C
C                        |    |              |           C
C                        |   /  -B           |           C
C                        |  /|G              |           C
C                        |/-|                |           C
C                         ------------------             C
C                                                        C
C                IV(I)=G*IE(I) FOR G*IE(I) GE B          C
C                              OR LE -B                  C
C                IV(I)=0 FOR G*IE(I) LT B AND GT -B      C
C     IE(I)-THE INPUT NODE NUMBER.                       C
C                                                        C
C     IV(I)-THE OUTPUT NODE NUMBER.                      C
C     G(I)-THE GAIN FIELD. G(I) REPRESENTS THE GAIN      C
C       OF THE BLOCK.                                    C
C                                                        C
C     P(I)-THE POLE FIELD.                               C
C       FOR ID(I)=2 OR 3 P(I)=POLE                       C
C       FOR ID(I)=4 P(I)=DAMPING FACTOR                  C
C       FOR ID(I)=5 P(I)=UPPER LIMIT,UL.                 C
C       FOR ID(I)=6 P(I)=REFERENCE,B.                    C
C                                                        C
C     Z(I)-THE ZERO FIELD                                C
C       FOR ID(I)=3 Z(I)=ZERO                            C
C       FOR ID(I)=4 Z(I)=NATURAL FREQUENCY,WN.           C
C       FOR ID(I)=5 Z(I)=LOWER LIMIT,LL.                 C
```

```
C                                                                      C
C CARDS 48-72 AMP(I),DELAY(I),INPUT(I),IEE(I)                          C
C                                                                      C
C         FORMAT(2F10.5,A4,I2)                                         C
C                                                                      C
C         ONE CARD IS NEEDED FOR EACH FORCING FUNCTION ON              C
C         THE SIMULATED COMPENSATED SYSTEM.                            C
C                                                                      C
C         AMP(I),DELAY(I),ANDINPUT(I) ARE AS DESCRIBED FOR             C
C         CARDS 17,19,AND 21,EXCEPT THEY APPLY TO THE COM-             C
C         PENSATED SYSTEM IN THIS CASE.                                C
C                                                                      C
C         IEE(I)-THE NODE TO WHICH THE FORCING FUNCTION IS             C
C         APPLIED.                                                     C
C                                                                      C
C CARDS 73-78 TITLE CARDS FOR VERSATEC PLOTS. IF NGRAPH=2,             C
C         NOUT*2 TITLE CARDS ARE REQUIRED(2 PER OUTPUT)                C
C         TITLES GO IN COLUMNS 1-48 ONLY.                              C
C                                                                      C
C                           *       *                                  C
C                           *  NOTE *                                  C
C                           *       *                                  C
C                                                                      C
C WHEN OPTIMIZING,ONE FORTRAN CARD PER FREE VARIABLE MUST              C
C BE INSERTED IN SUBROUTINE PLANT. THE APPROPRIATE POSITIONC
C IS IDENTIFIED BY THE COMMENT CARD ' ENTER G(I)=C(I) VAL-             C
C UES AT THIS POINT '.                                                 C
C                                                                      C
C**********************************************************************C
C                                                                      C
C THE FOLLOWING JCL CARDS SHOULD FOLLOW JOB CARD,EACH                  C
C BEGINNING IN COLUMN 1.                                               C
C                                                                      C
C   // EXEC FORTCLGV,REGION.GO=350K                                    C
C   //FORT.SYSPRINT DD DUMMY                                           C
C   //SYSIN DD *                                                       C
C                                                                      C
C THE SECOND OF THESE SHOULD BE REMOVED IF A PROGRAM                   C
C LISTING IS DESIRED AS WELL AS OUTPUT DATA.                           C
C                                                                      C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 XS(25),XL(25),XU(25),X(2),XDOT(2),DELTA(3),
     *W(3),WN(3),BETA(3),THACUT(3001,3),XDATA(3001,3),
     1AMP(3),DELAY(3)
      DIMENSION IOUT(3),IWT(3),ITAB(3),INPUT(3)
      INTEGER STEP,RAMP,PARA
      DATA STEP,RAMP,PARA/4HSTEP,4HRAMP,4HPARA/
      COMMON THAOUT,XDATA,T,DT,TF,
     *NOUT,IIN,NV,M3,ICONT,NEC,ISKIP,ITF,IOUT,NOPT
      COMMON /REG1/ W,IPM
C                                                                      C
C                    INPUT CONTROL FLAGS                               C
C                                                                      C
      READ(5,50) NRUNS,NOPT,NGRAPH,NREF,IIN,NOUT,IPRINT,IFREQ
      WRITE(6,51)NRUNS,IIN,NOUT
      IF(NOPT.EQ.1)WRITE(6,52)
      IF(NOPT.EQ.2)WRITE(6,53)
      IF(NGRAPH.EQ.1)WRITE(6,54)
      IF(NGRAPH.EQ.2)WRITE(6,55)
      IF(NGRAPH.EQ.3)WRITE(6,56)
      IF(NREF.EQ.1)WRITE(6,57)
      IF(NREF.EQ.2)WRITE(6,58)
      IF(IPRINT.EQ.1 )WRITE(6,60) IFREQ
      IF(IPRINT.EQ.2 )WRITE(6,59)
      DO 30 IRUN=1,NRUNS
        READ(5,61)T47,DT,TF
        ITF=TF/DT
        IF(ITF.GT.3001)ITF=3001
        IDP=ITF
        NPPLT=IDP/5
```

```
      WRITE(6,62)T47,DT,TF
C                                                               C
C           INPUT OPTIMIZATION DATA                             C
C                                                               C
      GO TO (1,2),NOPT
    1 READ(5,50)NV,NAV,NTA,NPR,IP,IPM
      READ(5,61)(XS(I),I=1,NV)
      READ(5,61)(XU(I),I=1,NV)
      READ(5,61)(XL(I),I=1,NV)
      WRITE(6,63)NV,NAV,NTA,NPR,IP,IPM
      DO 31 I=1,NV
        WRITE(6,64)I,XS(I),I,XU(I),I,XL(I)
   31 CONTINUE
    2 GO TO (3,4),NREF
    3 DO 32 I=1,NOUT
C                                                               C
C           INPUT/CALCULATE REFERENCE CURVES                    C
C                                                               C
        READ(5,50)IOUT(I),IWT(I),ITAB(I)
        W(I)=IWT(I)/100000.D0
        WRITE(6,65) I,IOUT(I),I, W (I)
        IF(ITAB(I).EQ.1)WRITE(6,76)I
        IF(ITAB(I).EQ.2)WRITE(6,77)I
        ITA=ITAB(I)
        GO TO (5,6),ITA
    5   READ(5,61)(XDATA(IDATA,I),IDATA=1,ITF)
        GO TO 32
    6   READ(5,75)BETA(I),DELTA(I),WN(I),AMP(I),DELAY(I),
     *  INPUT(I)
        WRITE(6,66)I,BETA(I),I,DELTA(I),I,WN(I),I,AMP(I),
     *  I,INPUT(I),I,DELAY(I)
        T=T47
        A17=0.D0
        IDATA=0
        X(1)=0.D0
        X(2)=0.D0
        NT=0
        DO 33 J=1,ITF
   11     IF(T.GE.DELAY(I).AND.INPUT(I).EQ.STEP)A17=AMP(I)
          IF(T.GE.DELAY(I).AND.INPUT(I).EQ.RAMP)A17=AMP(I)
     *    *(T-DELAY(I))
          IF(T.GE.DELAY(I).AND.INPUT(I).EQ.PARA)A17=AMP(I)
     *    *((T-DELAY(I))**2)
          XDOT(1)=X(2)
          XDOT(2)=-2.D0*DELTA(I)*WN(I)*X(2)-(WN(I)**2)*
     *    X(1)+BETA(I)*A17
          S=RKLDEQ(2,X,XDOT,T,DT,NT)
          IF(S-1.D0)10,11,16
   10     WRITE(6,67) I
          STOP
   16     XDATA(J,I)=X(1)
   33   CONTINUE
   32 CONTINUE
      GO TO 12
    4 DO 34 I=1,ITF
        DO 35 J=1,NOUT
          XDATA(I,J)=0.D0
   35   CONTINUE
   34 CONTINUE
   12 T=T47
      ISKIP=0
C                                                               C
C           IF SIMULATION ONLY CALL PLANT                       C
C                                                               C
      IF(NOPT.EQ.2) CALL PLANT(C)
      IF(NOPT.EQ.2) GO TO 13
C                                                               C
C                     OPTIMIZATION                              C
C                                                               C
      R=1.D0/3.D0
      WRITE(6,73)
      CALL BOXPLX(NV,NAV,NPR,NTA,R,XS,IP,XU,XL,YMN,IER)
```

```
      WRITE(6,73)
      WRITE(6,68)
      DO 36 I=1,NV
        WRITE(6,69)I, XS(I)
36    CONTINUE
      WRITE(6,70) YMN,IER
C                                                              C
C                        PRINTED OUTPUT                        C
C                                                              C
13    GO TO (14,15),IPRINT
14    WRITE(6,71)
      DO 37 I=1,ITF,IFREQ
        TP=DT*I
        WRITE(6,72)TP,(XDATA(I,J),THACUT(I,J),J=1,NOUT)
37    CONTINUE
C                                                              C
C                        PLOTTED OUTPUT                        C
C                                                              C
15    GO TO (7,8,9),NGRAPH
 7    DO 38 II=1,NOUT
        WRITE(6,73)
        CALL PPLT(NPPLT,IDP,II)
38    CONTINUE
      GO TO 9
 8    DO 39 II=1,NOUT
        WRITE(6,73)
        CALL PIC(NPPLT,IDP,II)
39    CONTINUE
 9    WRITE(6,74) IRUN
30  CONTINUE
    STOP
50  FORMAT(8I5)
51  FORMAT('0',I2,1X,'RUN(S) WILL BE MADE',4X,I2,1X,
   *'INPUTS',4X,I2,1X,'OUTPUTS')
52  FORMAT('0','OPTIMIZATION CALLED FOR')
53  FORMAT('0','SIMULATION ONLY CALLED FOR')
54  FORMAT('0','PRINTER PLOT(S) CALLED FOR')
55  FORMAT('0','VERSATEC PLOTS CALLED FOR')
56  FORMAT('0','NO GRAPHICAL OUTPUT CALLED FOR')
57  FORMAT('0','REFERENCE CURVE CALLED FOR')
58  FORMAT('0','NO REFERENCE CURVE CALLED FOR')
59  FORMAT('0','NO PRINTED SIMULATION DATA CALLED FOR')
60  FORMAT('0','SIMULATION DATA PRINT FREQUENCY=',1X,I2)
61  FORMAT(8F10.5)
62  FORMAT('0','INITIAL TIME=',F10.5,3X,'STEP SIZE=',F10.5
   *,3X,'FINAL TIME=',F10.5)
63  FORMAT('0','NV=',I2,3X,'NAV=',I2,3X,'NTA=',I5,3X,'NPR=
   *',I5,3X,'IP=',I1,3X,'IPM=',I1)
64  FORMAT('0','XS(',I2,')=',F10.5,5X,'XU(',I2,')=',F10.5,
   *5X,'XL(',I2,')=',F10.5)
65  FORMAT('0','IOUT(',I1,')=',I2,5X,'W(',I1,')=',F10.5)
66  FORMAT('0','BETA(',I1,')=',F10.5,3X,'DELTA(',I1,')=',
   *F10.5,3X,'WN(',I1,')=',F10.5,3X,'AMP(',I1,')=',F10.5,
   *3X,'INPUT(',I1,')=',  A4 ,3X,'DELAY(',I1,')=',F10.5)
67  FORMAT('0','INTEGRATION TROUBLE-REFERENCE CURVE')
68  FORMAT('0','THE FREE PARAMETERS ASSOCIATED WITH THE
   *MINIMUM PERFORMANCE MEASURE ARE:')
69  FORMAT('0','XS(',I2,')=',F20.10)
70  FORMAT('0','THE MINIMUM PERFORMANCE MEASURE IS:',2X,
   *F20.10,5X,'IER=',2X,I1)
71  FORMAT('1',3X,'TIME', 9X,'XDATA(1)', 7X,'THAOUT(1)',
   * 6X,'XDATA(2)', 7X,'THAOUT(2)', 6X,'XDATA(3)', 7X,'
   *THAOUT(3)')
72  FORMAT('0',6(F10.5,5X),F10.5)
73  FORMAT('1')
74  FORMAT('0','RUN NUMBER',1X,I1)
75  FORMAT(5F10.5,A4)
76  FORMAT('0','REF.CURVE(',I1,')=','TABULATED DATA')
77  FORMAT('0','REF. CURVE(',I1,')=SECOND ORDER RESPONSE')
    END
```

```
      SUBROUTINE PLANT(C)
C                                                                    C
C        SUBROUTINE PLANT(C) SIMULATES THE SYSTEM                    C
C                                                                    C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8  G(25),P(25),Z(25),DRIVE(25),THA(25),OMG(25),
     1THADOT(25),OMGDOT(25),DRVIN(25),FLAG(25,25),
     2 X2(25), X2DOT(25),  THAOUT(3001,3),
     3XDATA(3001,3),C(25),AMP(25),DELAY(25),TAG(25,25)
      DIMENSION IC(25),ID(25),IE(25),NF(25),IR(25),IV(25),
     1IEE(25),INPUT(25),IOUT(3)
      INTEGER STEP,RAMP,PARA
      DATA STEP,RAMP,PARA/4HSTEP,4HRAMP,4HPARA/
      COMMON THAOUT,XDATA,T,DT,TF,
     CNCUT,IIN,NV,M3,ICONT,NEQ,ISKIP,ITF,IOUT,NOPT
      IF (ISKIP-1) 1,5,5
    1 ISKIP = 2
      READ(5,24) N
      H1 = DT
      H2 = 0.5D0*H1
      N11 = 0
      N55 = 0
      N66 = 0
      ICK = 2*N
      WRITE(6,310)
      DO 3 I=1,N
C                                                                    C
C                         INPUT BLOCK DATA                           C
C                                                                    C
         READ(5,25)IC(I),ID(I),IE(I),IV(I),G(I),P(I),Z(I)
         IF(ID(I).EQ.1) N11=N11+1
         IF( ID(I).EQ.5) N55=N55+1
         IF(ID(I).EQ.6) N66=N66+1
         WRITE(6,26) IC(I),ID(I),IE(I),IV(I),G(I),P(I),Z(I)
    3 CONTINUE
         WRITE(6,312)
         DO 316 I=1,IIN
C                                                                    C
C                      INPUT FORCING FUNCTIONS                       C
C                                                                    C
         READ(5,311)AMP(I),DELAY(I),INPUT(I),IEE(I)
           WRITE(6,317)IEE(I),AMP(I),INPUT(I),DELAY(I)
  316      CONTINUE
      N11 = 4*N11
      N55 = 4*N55
      N66 = 4*N66
      NEQ = N-1
C                                                                    C
C                            SET POINTERS                            C
C                                                                    C
      DO 4 J=1,N
        DO 4 K=1,N
           FLAG(K,J)=0.D0
           IF(IV(K).EQ.IE(J))FLAG(K,J)=1.D0
    4 CONTINUE
C                                                                    C
C                           INITIALIZATION                           C
C                                                                    C
    5 DO 6 ICLR=1,N
         THA(ICLR)=0.D0
         THADOT(ICLR)=0.D0
         OMG(ICLR)=0.D0
         OMGDOT(ICLR)=0.D0
         X2(ICLR)=0.D0
         X2DOT(ICLR)=0.D0
    6 CONTINUE
         T=0.D0
      NR2 = 1
      NR3 = 1
      NR4 = 1
      M3 = 0
      ICONT = 0
```

```
      IWAIT = 0
      IT = 0
      ILAST = 0
      I5LAST = 0
      I6LAST = 0
      IF(NOPT.EQ.2) GO TO 7
C                                                                     C
C     ENTER C(I)=G(I) VALUES AT THIS POINT                            C
C                                                                     C
    7 DO 9 MDRV=1,N
        DRIVE(MDRV)=0.D0
        DRVIN(MDRV)=0.D0
        DO 8 M=1,N
          IF(IV(M).NE.IE(MDRV)) GO TO 315
          DRIVE(MDRV)=DRIVE(MDRV)+THA(M)*FLAG(M,MDRV)
  315     IF(IEE(M).NE.IE(MDRV)) GO TO 8
          IF(INPUT(M).EQ.STEP) GO TO 32
          IF(INPUT(M).EQ.RAMP) GO TO 33
          IF(INPUT(M).EQ.PARA) GO TO 34
   32     IF(T.GE.DELAY(M))DRVIN(MDRV)=AMP(M)
          GO TO 8
   33     IF(T.GE.DELAY(M))DRVIN(MDRV)=AMP(M)*(T-DELAY(M))
          GO TO 8
   34     IF(T.GE.DELAY(M))DRVIN(MDRV)=AMP(M)*((T-DELAY(M))
     *    **2)
    8     CONTINUE
        DRIVE(MDRV)=DRIVE(MDRV)+DRVIN(MDRV)
    9   CONTINUE
   10 M3 = M3+1
C                                                                     C
C                      BLOCK SIMULATION                               C
C                                                                     C
      IF (IWAIT.EQ.0) T = T+H2
      IF (ID(M3).EQ.1) GO TO 11
      IF (ID(M3).EQ.2) GO TO 12
      IF (ID(M3).EQ.3) GO TO 13
      IF (ID(M3).EQ.4) GO TO 14
      IF (ID(M3).EQ.5) GO TO 15
      IF (ID(M3).EQ.6) GO TO 16
      WRITE (6,28)
      STOP
   11 THA(M3) = G(M3)*DRIVE(M3)
      IWAIT = IWAIT+1
      ILAST = ILAST+1
      IF ((ILAST.EQ.N11).AND.(ICONT.EQ.NEQ)) GO TO 21
      GO TO 18
   12 THADOT(M3) = -P(M3)*THA(M3)+G(M3)*DRIVE(M3)
      S = RKLDE2(THA,THADOT,NR2)
      IWAIT = IWAIT+1
      IF(S-1.D0)17,18,21
   13 OMGDOT(M3) = -P(M3)*OMG(M3)+G(M3)*DRIVE(M3)
      S = RKLDE3(OMG,OMGDOT,NR3)
      THA(M3) = (Z(M3)-P(M3))*OMG(M3)+G(M3)*DRIVE(M3)
      IWAIT = IWAIT+1
      IF(S-1.D0)17,18,21
   14 V = CCPLX(P,Z,G,DRIVE,X2,OMG,NR4)
      THA(M3) = OMG(M3)
      IWAIT = IWAIT+1
      IF(V-1.D0)17,18,21
   15 THA(M3) = G(M3)*DRIVE(M3)
      IF (THA(M3).GT.P(M3)) THA(M3)=P(M3)
      IF (THA(M3).LT.Z(M3)) THA(M3)=Z(M3)
      IWAIT = IWAIT+1
      I5LAST = I5LAST+1
      IF ((I5LAST.EQ.N55).AND.(ICONT.EQ.NEQ)) GO TO 21
      GO TO 18
   16 THA(M3) = G(M3)*DRIVE(M3)
      IF((DABS(THA(M3))).LT.P(M3))THA(M3)=0.D0
      IWAIT = IWAIT+1
      I6LAST = I6LAST+1
      IF ((I6LAST.EQ.N66).AND.(ICONT.EQ.NEQ)) GO TO 21
      GO TO 18
```

```
   17 WRITE (6,29)
      STOP
   18 ICONT = ICONT+1
      IF (ICONT-N) 10,19,20
   19 NR2 = NR2+1
      NR3 = NR3+1
      NR4 = NR4+1
      M3 = 0
      ICONT = 0
      IF (IWAIT.EQ.ICK) IWAIT=0
      GO TO 7
   20 WRITE (6,30)
      STOP
C                                                                      C
C                        STORE OUTPUT DATA                             C
C                                                                      C
   21 IT = IT+1
      DO 38 I=1,NOUT
         THAOUT(IT,I)=THA(IOUT(I))
   38 CONTINUE
      IF (T-TF) 22,22,23
   22 NR2 = 1
      NR3 = 1
      NR4 = 1
      M3 = 0
      ICONT = 0
      IWAIT = 0
      ILAST = 0
      I5LAST = 0
      I6LAST = 0
      GO TO 7
   23 RETURN
   24 FORMAT(I2)
   25 FORMAT(4I2, 2X,3F10.5)
   26 FORMAT (/,5H BLK ,I2,I1,2H =,2I2,6X,3E20.7)
   27 FORMAT (//, 2X,'THETA OUT IS THA(',I2,')')
   28 FORMAT (40H *** EQN SWITCH CONTROL DID NOT WORK ***)
   29 FORMAT (20H INTEGRATION TROUBLE)
   30 FORMAT(58X,'ERROR IN INTERGATION.',2X,
     1'ATTEMPTED TO INTEG. MORE THAN N-EQTN')
  310 FORMAT('0',40X,'THE SYSTEM BLOCKS ARE:')
  311 FORMAT(2F10.5,A4,I2)
  312 FORMAT('0',40X,'THE SYSTEM FORCING FUNCTIONS ARE:')
  317 FORMAT('0','DRVIN(',I2,')=',F10.5,A4,F10.5)
      END

      FUNCTION RKLDEQ(N,X,XDOT,T,DT,NT)
C                                                                      C
C                 CALCULATES RESPONSE OF SECOND                        C
C                   ORDER REFERENCE CURVE                              C
C                                                                      C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 X(2),XDOT(2),Q(25)
      NT = NT + 1
      GO TO (1,2,3,4),NT
    1 H1 = DT
      H2 = H1*0.5D0
      H3 = H1 *2.0D0
      H6 = H1/6.0D0
      DO 11 J = 1,N
   11 Q(J) = 0.D0
      A = 0.5D0
      T = T + H2
      GO TO 5
    2 A = 0.29289321881345 25
      GO TO 5
    3 A = 1.7071067811865475
      T = T + H2
      GO TO 5
    4 DO 41 I = 1,N
   41 X(I) = X(I) + H6*XDOT(I) - Q(I)/3.D0
```

```
      NT = 0
      RKLDEQ = 2.
      GO TO 6
    5 DO 51 L = 1,N
      X(L) = X(L) + A*(DT*XDOT(L)-Q(L))
   51 Q(L) = H3*A*XDOT(L) + (1.D0 - 3.D0*A)*Q(L)
      RKLDEQ = 1.
    6 RETURN
      END

      FUNCTION RKLDE2 (X,XDOT,NR2)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 THAOUT(3001,3),XDATA(3001,3)
      REAL*8 X(4), XDOT(4), Q(25)
      DIMENSION IOUT(3)
      COMMON THAOUT,XDATA,T,DT,TF,
     CNOUT,IIN,NV,M3,ICONT,NEQ,ISKIP,ITF,IOUT
      GO TO (1,2,3,4), NR2
    1 H1 = DT
      H2 = H1*0.5D0
      H3 = H1*2.0D0
      H6 = H1/6.0D0
      Q(M3) = 0.D0
      A = 0.5D0
      GO TO 5
    2 A = 0.2928932188134525
      GO TO 5
    3 A = 1.7071067811865475
      GO TO 5
    4 X(M3) = X(M3)+H6*XDOT(M3)-Q(M3)/3.D0
      RKLDE2 = 1.
      IF (ICONT.EQ.NEQ) RKLDE2=2.
      GO TO 6
    5 X(M3) = X(M3)+A*(DT*XDOT(M3)-Q(M3))
      Q(M3) = H3*A*XDOT(M3)+(1.D0-3.D0*A)*Q(M3)
      RKLDE2 = 1.
    6 RETURN
      END

      FUNCTION RKLDE3 (X,XDOT,NR3)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 THAOUT(3001,3),XDATA(3001,3)
      REAL*8 X(4), XDOT(4), Q(25)
      DIMENSION IOUT(3)
      COMMON THAOUT,XDATA,T,DT,TF,
     CNOUT,IIN,NV,M3,ICONT,NEQ,ISKIP,ITF,IOUT
      GO TO (1,2,3,4), NR3
    1 H1 = DT
      H2 = H1*0.5D0
      H3 = H1*2.0D0
      H6 = H1/6.0D0
      Q(M3) = 0.D0
      A = 0.5D0
      GO TO 5
    2 A = 0.29289321881345 25
      GO TO 5
    3 A = 1.7071067811865475
      GO TO 5
    4 X(M3) = X(M3)+H6*XDOT(M3)-Q(M3)/3.D0
      RKLDE3 = 1.
      IF (ICONT.EQ.NEQ) RKLDE3=2.
      GO TO 6
    5 X(M3) = X(M3)+A*(DT*XDOT(M3)-Q(M3))
      Q(M3) = H3*A*XDOT(M3)+(1.D0-3.D0*A)*Q(M3)
      RKLDE3 = 1.
    6 RETURN
      END

      FUNCTION CCPLX (P,Z,G,DRIVE,X2,OMG,NR4)
```

```
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 THAOUT(3001,3),XDATA(3001,3),OMGDOT(1)
      REAL*8 OMG(1),P(1),Z(1),G(1),DRIVE(1),X2(25),X2DOT(25)
      DIMENSION IOUT(3)
      COMMON THAOUT,XDATA,T,DT,TF,
     CNOUT,IIN,NV,M3,ICONT,NEQ,ISKIP,ITF,IOUT
      OMGDOT(M3) = X2(M3)
      SS = RKLDE3(OMG,OMGDOT,NR4)
      X2DOT(M3) = -2.*P(M3)*Z(M3)*X2(M3)-Z(M3)**2*OMG(M3)+
     1G(M3)*DRIVE(M3)
      SSS = RKLDE4(X2,X2DOT,NR4)
      CCPLX = SSS
      RETURN
      END

      FUNCTION RKLDE4 (X,XDOT,NR4)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 THAOUT(3001,3),XDATA(3001,3)
      REAL*8 X(1), XDOT(1), QC(25)
      DIMENSION IOUT(3)
      COMMON THAOUT,XDATA,T,DT,TF,
     CNOUT,IIN,NV,M3,ICONT,NEQ,ISKIP,ITF,IOUT
      GO TO (1,2,3,4), NR4
    1 H1 = DT
      H2 = H1*0.5D0
      H3 = H1*2.0D0
      H6 = H1/6.0D0
      QC(M3) = 0.D0
      A = 0.5D0
      GO TO 5
    2 A = 0.2928932188134525
      GO TO 5
    3 A = 1.7071067811865475
      GO TO 5
    4 X(M3) = X(M3)+H6*XDOT(M3)-QC(M3)/3.D0
      RKLDE4 = 1.
      IF (ICONT.EQ.NEQ) RKLDE4=2.
      GO TO 6
    5 X(M3) = X(M3)+A*(DT*XDOT(M3)-QC(M3))
      QC(M3) = H3*A*XDOT(M3)+(1.D0-3.D0*A)*QC(M3)
      RKLDE4 = 1.
    6 RETURN
      END

      FUNCTION KE(C)
C                                                             C
C         EVALUATES IMPLICIT CONSTRAINTS FOR BOXPLX           C
C                                                             C
      REAL*8 C(25)
      KE=0
      RETURN
      END

      FUNCTION FE(C)
C                                                             C
C           THE FUNCTION MINIMIZED BY BOXPLX                  C
C                                                             C
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 THAOUT(3001,3),XDATA(3001,3),C(25),W(3)
      COMMON THAOUT,XDATA,T,DT,TF,
     *NOUT,IIN,NV,M3,ICONT,NEQ,ISKIP,ITF,IOUT
      COMMON /REG1/ W,IPM
      DO 10 I=1,NOUT
        THAOUT(1,I)=0.D0
   10 CONTINUE
      DIFF=0.D0
      PI=0.D0
      CALL PLANT(C)
      T=DT
```

```
      DO 11  I=1,ITF
        DO 12  J=1,NOUT
          DIFF=XDATA(I,J)-THAOUT(I,J)
          GO TO(1,2,3,4),IPM
 1        PI=PI+W(J)*(DIFF*2)
          GO TO 12
 2        PI=PI+W(J)*(DIFF**2)*T
          GO TO 12
 3        PI=PI+W(J)*DABS(DIFF)
          GO TO 12
 4        PI=PI+W(J)*DABS(DIFF)*T
12      CONTINUE
        T=T+DT
11    CONTINUE
      FE=PI
      RETURN
      END

      SUBROUTINE PPLT(NPPLT,IDP,II)
C                                                              C
C         AUTOSCALES AND CALLS FOR PRINTR PLOTS                C
C                                                              C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 XX(900),YY(900),WW(900)
      REAL*4  TX(4),TY(4)
      REAL*8 THAOUT(3001,3),XDATA(3001,3)
      DIMENSION IOUT(3)
      COMMON THAOUT,XDATA,T,DT,TF,
     CNOUT,IIN,NV,M3,ICONT,NEC,ISKIP,ITF,IOUT
      DO 1  I=1,900
      XX(I)  = 0.
      YY(I)  = 0.
    1 WW(I)  = 0.
      T=0.D0
      TSTEP=5.D0*DT
      J = 0
      BIGX=0.D0
      BIGY=0.D0
      SMLX=0.D0
      SMLY=0.D0
      DO 2  I=1,IDP,5
      J = J+1
      XX(J)  = T
      YY(J)=XDATA(I,II)
      WW(J)=THAOUT(I,II)
      X  = T
      XD =  YY(J)
      TH =  WW(J)
      YMAX=DMAX1(XD,TH)
      YMIN=DMIN1(XD,TH)
      XMAX=DMAX1(BIGX,X)
      IF (BIGX.LT.X) BIGX=X
      IF (BIGY.LT.YMAX) BIGY=YMAX
      IF (SMLY.GT.YMIN)  SMLY=YMIN
      T  =  T+TSTEP
    2 CONTINUE
      TX(1)  =  0.
      TX(2)  =  0.
      TX(3)  =  SMLX
      TX(4)  =  BIGX
      TY(1)  =  BIGY
      TY(2)  =  SMLY
      TY(3)  =  0.
      TY(4)  =  0.
      WRITE  (6,3) BIGX,BIGY,SMLY
      CALL  PLOTP  (TX,TY,4,1)
      CALL  PLOTP  (XX,WW,NPPLT,2)
      CALL  PLOTP  (XX,YY,NPPLT,3)
      WRITE(6,4)
      IF(IDP.EQ.4500) WRITE(6,5)
      RETURN
```

```
    3 FORMAT(2X,'BIGX= ',E15.7,2X,'BIGY= ',E15.7,2X,
     1'SMLY= ',E15.7)
    4 FORMAT(//,2X,'SYSTEM RESPONSE FOR PROBLEM -----')
    5 FORMAT(//,2X,'STOP AT 900 GRAPH POINTS')
      END

      SUBROUTINE PIC (NPPLT,IDP,II)
C                                                                    C
C        AUTOSCALES AND CALLS FOR VERSATEC PLOTS                     C
C                                                                    C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*4 XX(900),YY(900),WW(900)
      REAL*4TX(4),TY(4)
      REAL*8 THAOUT(3001,3),XDATA(3001,3),TITLE(12)
      REAL *8LABC/'        '/
      REAL *4LABA/' A  '/,LABD/'  D '/
      DIMENSION IOUT(3)
      COMMON THAOUT,XDATA,T,DT,TF,
     *NOUT,IIN,NV,M3,ICONT,NEC,ISKIP,ITF,IOUT
      READ (5,2) TITLE
      T=0.D0
      TSTEP=5.D0*DT
      J = 0
      BIGX = 0.D0
      BIGY = 0.D0
      SMLX = 0.D0
      SMLY = 0.D0
      DO 1 I=1,IDP,5
      J = J+1
      XX(J) = T
      YY(J)=XDATA(I,II)
      WW(J)=THAOUT(I,II)
      X = T
      XD = YY(J)
      TH = WW(J)
      YMAX = DMAX1(XD,TH)
      YMIN = DMIN1(XD,TH)
      XMAX = DMAX1(BIGX,X)
      IF (BIGX.LT.X) BIGX=X
      IF (BIGY.LT.YMAX) BIGY=YMAX
      IF (SMLY.GT.YMIN) SMLY=YMIN
      T = T+TSTEP
    1 CONTINUE
      TX(1) = 0.
      TX(2) = 0.
      TX(3) = SMLX
      TX(4) = BIGX
      TY(1) = BIGY
      TY(2) = SMLY
      TY(3) = 0.
      TY(4) = 0.
      CALL DRAW (4,TX,TY,1,1,LABC,TITLE,0,0,0,0,0,0,8,8,0,L)
      CALL DRAW (NPPLT,XX,WW,2,0,LABA,TITLE,0,0,0,0,0,0,8,8,
     *0,L)
      CALL DRAW (NPPLT,XX,YY,3,0,LABD,TITLE,0,0,0,0,0,0,8,8,
     *0,L)
      IF (L.NE.0) WRITE (6,3) L
      RETURN
    2 FORMAT (6A8)
    3 FORMAT (//,'  GRAPH NOT COMPLETED. OUTPUT CODE = ',I2)
      END

      SUBROUTINE BOXPLX (NV,NAV,NPR,NTZ,RZ,XS,IP,BU,BL,YMN,
     *IER)
C ******************************************************************  C
C                                                                    C
C     SUBROUTINE BOXPLX          (CATEGORY H0)                       C
C                                                                    C
C     PURPOSE                                                        C
C                                                                    C
```

```
C     BOXPLX IS A SUBROUTINE USED TO SOLVE THE PROBLEM           C
C     OF LOCATING A MINIMUM (OR MAXIMUM) OF AN ARBITRARY         C
C     OBJECTIVE FUNCTION SUBJECT TO ARBITRARY EXPLICIT           C
C     AND/OR IMPLICIT CONSTRAINTS BY THE COMPLEX METHOD          C
C     OF M.J. BOX. EXPLICIT CONSTRAINTS ARE DEFINED AS           C
C     UPPER AND LOWER BOUNDS ON THE INDEPENDENT VARIABL-         C
C     ES. IMPLICIT CONSTRAINTS MAY BE ARBITRARY FUNCTIONS        C
C     OF THE VARIABLES. TWO FUNCTION SUBPROGRAMS TO              C
C     EVALUATE THE OBJECTIVE FUNCTION AND IMPLICIT CON-          C
C     STRAINTS, RESPECTIVELY, MUST BE SUPPLIED BY THE            C
C     USER (SEE EXAMPLE BELOW). BOXPLX ALSO HAS THE OPT-         C
C     ION TO PERFORM INTEGER PROGRAMMING, WHERE THE VAL-         C
C     UES OF THE INDEPENDENT VARIABLES ARE RESTRICTED TO         C
C     INTEGERS.                                                  C
C                                                                C
C  USAGE                                                         C
C                                                                C
C     CALL BOXPLX (NV,NAV,NPR,NTA,R,XS,IP,XU,XL,YMN,IER)         C
C                                                                C
C  DESCRIPTION OF PARAMETERS                                     C
C                                                                C
C     NV     AN INTEGER INPUT DEFINING THE NUMBER OF INDE-       C
C            PENDENT VARIABLES OF THE OBJECTIVE FUNCTION         C
C            TO BE MINIMIZED. NOTE: MAXIMUM NV + NAV IS          C
C            PRESENTLY 50. MAXIMUM NV IS 25. IF THESE            C
C            LIMITS MUST BE EXCEEDED, PUNCH A SOURCE DECK        C
C            IN THE USUAL MANNER, AND CHANGE THE DIMEN-          C
C            SION STATEMENTS.                                    C
C                                                                C
C     NAV    AN INTEGER INPUT DEFINING THE NUMBER OF AUXI-       C
C            LIARY VARIABLES THE USER WISHES TO DEFINE FOR       C
C            HIS OWN CONVENIENCE. TYPICALLY HE MAY WISH          C
C            TO DEFINE THE VALUE OF EACH IMPLICIT CON-           C
C            STRAINT FUNCTION AS AN AUXILIARY VARIABLE. IF       C
C            THIS IS DONE, THE OPTIONAL OUTPUT FEATURE           C
C            OF BOXPLX CAN BE USED TO OBSERVE THE VALUES         C
C            OF THOSE CONSTRAINTS AS THE SOLUTION PRO-           C
C            GRESSES. AUXILIARY VARIABLES, IF USED, SHOULD       C
C            BE EVALUATED IN FUNCTION KE (DEFINED BELOW).        C
C            NAV MAY BE ZERO.                                    C
C                                                                C
C     NPR    INPUT INTEGER CONTROLLING THE FREQUENCY OF          C
C            OUTPUT DESIRED FOR DIAGNOSTIC PURPOSES. IF          C
C            NPR .LE. 0, NO OUTPUT WILL BE PRODUCED BY           C
C            BOXPLX. OTHERWISE, THE CURRENT COMPLEX OF           C
C            K=2*NV VERTICES AND THEIR CENTROID WILL BE          C
C            OUTPUT AFTER EACH NPR PERMISSIBLE TRIALS. THE       C
C            NUMBER OF TOTAL TRIALS, NUMBER OF FEASIBLE          C
C            TRIALS, NUMBER OF FUNCTION EVALUATIONS AND          C
C            NUMBER OF IMPLICIT CONSTRAINT EVALUATIONS ARE       C
C            INCLUDED IN THE OUTPUT.                             C
C            ADDITIONALLY, (WHEN NPR .GT. 0) THE SAME IN-        C
C            FORMATION WILL BE OUTPUT:                           C
C                                                                C
C              1) IF THE INITIAL POINT IS NOT FEASIBLE.          C
C              2) AFTER THE FIRST COMPLETE COMPLEX IS            C
C                 GENERATED.                                     C
C              3) IF A FEASIBLE VERTEX CANNOT BE FOUND AT        C
C                 SOME TRIAL.                                    C
C              4) IF THE OBJECTIVE VALUE OF A VERTEX CAN-        C
C                 NOT BE MADE NO-LONGER-WORST.                   C
C              5) IF THE LIMIT ON TRIALS (NTA) IS REACHED        C
C                 AND,                                           C
C              6) WHEN THE OBJECTIVE FUNCTION HAS BEEN UN-       C
C                 CHANGED FOR 2*NV TRIALS, INDICATING A          C
C                 LOCAL MINIMUM HAS BEEN FOUND.                  C
C                                                                C
C            IF THE USER WISHES TO TRACE THE PROGRESS OF         C
C            A SOLUTION, A CHOICE OF NPR= 25, 50 OR 100 IS       C
C            RECOMMENDED.                                        C
C                                                                C
C     NTA    INTEGER INPUT OF LIMIT ON THE NUMBER OF             C
```

```
         XL(I) ARE THE LOWER LIMITS ON THE FREE PARA-
         METERS. OMIT IF NOPT =2.
```



```
         TRIALS ALLOWED IN THE CALCULATION.  IF THE
         USER INPUTS NTA .LE. 0, A DEFAULT VALUE OF
         2000 IS USED.  WHEN THIS LIMIT IS REACHED
         CONTROL RETURNS TO THE CALLING PROGRAM WITH
         THE BEST ATTAINED OBJECTIVE FUNCTION VALUE IN
         YMN, AND THE BEST  ATTAINED SOLUTION POINT IN
         XS.

   R     A REAL NUMBER INPUT TO DEFINE THE FIRST RAN-
         DOM NUMBER USED IN DEVELOPING THE INI-
         TIAL COMPLEX OF 2*NV VERTICES.
         (0. GT. R .LT. 1.) IF R IS NOT WITHIN THESE
         BOUNDS, IT WILL BE REPLACED BY 1./3. .

   XS    INPUT REAL ARRAY DIMENSIONED AT LEAST NV+NAV.
         THE FIRST NV MUST CONTAIN A FEASIBLE ORIGIN
         FOR STARTING THE CALCULATION.  THE LAST NAV
         NEED NOT BE INITIALIZED.  UPON RETURN FROM
         BOXPLX, THE FIRST NV ELEMENTS OF THE ARRAY
         CONTAIN THE COORDINATES OF THE MINIMUM OB-
         JECTIVE FUNCTION, AND THE REMAINING NAV
         (NAV .GE. 0) CONTAIN THE VALUES OF THE
         CORRESPONDING AUXILIARY VARIABLES.

   IP    INTEGER INPUT FOR OPTIONAL INTEGER PRO-
         GRAMMING.  IF IP=1, THE VALUES OF THE INDE-
         PENDENT VARIABLES WILL BE REPLACED WITH
         INTEGER VALUES (STILL STORED AS REAL*4).

   XU    A REAL ARRAY DIMENSIONED AT LEAST NV INPUT-
         TING THE UPPER BOUND ON EACH INDEPENDENT
         VARIABLE, (EACH EXPLICIT CONSTRAINT).  INPUT
         VALUES ARE SLIGHTLY ALTERED BY BOXPLX.

   XL    A REAL ARRAY DIMENSIONED AT LEAST NV INPUT-
         TING THE LOWER BOUND ON EACH INDEPENDENT
         VARIABLE, (EACH EXPLICIT CONSTRAINT).  NOTE:
         FOR BOTH XU AND XL CHOOSE REASONABLE VALUES
         IF NONE ARE GIVEN, NOT VALUES WHICH ARE MAG-
         NITUDES ABOVE OR BELOW THE EXPECTED SOLUTION.
         INPUT VALUES ARE SLIGHTLY ALTERED BY BOXPLX.

   YMN   THIS OUTPUT IS THE VALUE (REAL*4) OF THE OB-
         JECTIVE FUNCTION, CORRESPONDING TO THE SOLU-
         TION POINT OUTPUT IN XS.

   IER   INTEGER ERROR RETURN.  TO BE INTERROGATED
         UPON RETURN FROM BOXPLX.  IER WILL BE ONE OF
         THE FOLLOWING:

            =-1  CANNOT FIND FEASIBLE VERTEX OR FEAS-
                 IBLE CENTROID AT THE START OR A RE-
                 START (SEE 'METHOD' BELOW).
            =0   FUNCTION VALUE UNCHANGED FOR 'N'
                 TRIALS.  (WHERE N=6*NV+10)  THIS IS
                 THE NORMAL RETURN PARAMETER.
            =1   CANNOT DEVELOP FEASIBLE VERTEX.
            =2   CANNOT DEVELOP A NO-LONGER-WORST VER-
                 TEX.
            =3   LIMIT ON TRIALS REACHED.  (NTA EX-
                 CEEDED)
            NOTE:  VALID RESULTS MAY BE RETURNED IN
                   ANY OF THE ABOVE CASES.

EXAMPLE OF USAGE

   THIS EXAMPLE MINIMIZES THE OBJECTIVE FUNCTION
   SHOWN IN THE EXTERNAL FUNCTION FE(X).  THERE ARE
   TWO INDEPENDENT VARIABLES X(1) & X(2), AND TWO IM-
   PLICIT CONSTRAINT FUNCTIONS X(3) & X(4) WHICH ARE
   EVALUATED AS AUXILIARY VARIABLES (SEE EXTERNAL
   FUNCTION KE(X) ).
```

```
C
C         DIMENSION  XS(4),XU(2),XL(2)
C
C      STARTING GUESS
          XS(1) = 1.0
          XS(2) = 0.5
C      UPPER LIMITS
          XU(1) = 6.0
          XU(2) = 6.0
C      LOWER LIMITS
          XL(1) = 0.0
          XL(2) = 0.0
C
          R = 9./13.
          NTA = 5000
          NPR = 50
          NAV = 2
          NV = 2
          IP = 0
C
          CALL BOXPLX (NV,NAV,NPR,NTA,R,XS,IP,XU,XL,YMN,IER)
          WRITE(6,1) ((XS(I),I=1,4),YMN,IER)
        1 FORMAT(////,'    THE POINT IS LOCATED AT (XS(I)=) ',
         *4(E13.7,5X),//,'    AND THE FUNCTION VALUE IS ',
         *E13.7,' IER = ',I5)
          STOP
          END
C
C
C         FUNCTION KE(X)
C
C      EVALUATE CONSTRAINTS. SET KE=0 IF NO IMPLICIT
C      CONSTRAINT IS VIOLATED, OR SET KE=1 IF ANY IMPLICIT
C      CONSTRAINT IS VIOLATED.
C
          DIMENSION X(4)
          X1 = X(1)
          X2 = X(2)
          KE = 0
          X(3) = X1 + 1.732051*X2
          IF (X(3) .LT. 0. .OR. X(3) .GT. 6.) GO TO 1
          X(4) = X1/1.732051 -X2
          IF (X(4) .GE. 0.)  RETURN
C
        1 KE = 1
          RETURN
          END
C
C
C         FUNCTION  FE(X)
C         DIMENSION X(4)
C
C      THIS IS THE OBJECTIVE FUNCTION.
          FE= -(X(2)**3 *(9.-(X(1)-3.)**2)/(46.76538))
          RETURN
          END
C
C         METHOD
C
C            THE COMPLEX METHOD IS AN EXTENSION AND ADAPTION OF
C            THE SIMPLEX METHOD OF LINEAR PROGRAMMING.  START-
C            ING WITH ANY ONE FEASIBLE POINT IN N-DIMENSION
C            SPACE A "COMPLEX" OF 2*N VERTICES IS CONSTRUCTED
C            BY SELECTING RANDOM POINTS WITHIN THE FEASIBLE
C            REGION.  FOR THIS PURPOSE N COORDINATES ARE FIRST
C            RANDOMLY CHOSEN WITHIN THE SPACE BOUNDED BY EX-
C            PLICIT CONSTRAINTS.  THIS DEFINES A TRIAL INITIAL
C            VERTEX. IT IS THEN CHECKED FOR POSSIBLE VIOLATION
C            OF IMPLICIT CONSTRAINTS.  IF ONE OR MORE ARE VIO-
C            LATED, THE TRIAL INITIAL VERTEX IS DISPLACED HALF
C            OF ITS DISTANCE FROM THE CENTROID OF PREVIOUSLY
C            SELECTED INITIAL VERTICES.  IF NECESSARY THIS DIS-
```

```
C      PLACEMENT PROCESS IS REPEATED UNTIL THE VERTEX HAS C
C      BECOME FEASIBLE.  IF THIS FAILS TO HAPPEN AFTER    C
C      5*N+10 DISPLACEMENTS, THE SOLUTION IS ABANDONED.   C
C      AFTER EACH VERTEX IS ADDED TO THE COMPLEX, THE     C
C      CURRENT CENTROID IS CHECKED FOR FEASIBILITY.  IF   C
C      IT IS INFEASIBLE, THE LAST TRIAL VERTEX IS ABAND-  C
C      ONED AND AN EFFORT TO GENERATE AN ALTERNATIVE      C
C      TRIAL VERTEX IS MADE.  IF 5*N+10 VERTICES ARE      C
C      ABANDONED CONSECUTIVELY, THE SOLUTION IS TER-      C
C      MINATED.                                           C
C                                                         C
C      IF AN INITIAL COMPLEX IS ESTABLISHED, THE BASIC    C
C      COMPUTATION LOOP IS INITIATED.  THESE INSTRUCTIONS C
C      FIND THE CURRENT WORST VERTEX, THAT IS, THE VERTEX C
C      WITH THE LARGEST CORRESPONDING VALUE FOR THE OB-   C
C      JECTIVE FUNCTION, AND REPLACE THAT VERTEX BY ITS   C
C      OVER-REFLECTION THROUGH THE CENTROID OF ALL OTHER  C
C      VERTICES.  (IF THE VERTEX TO BE REPLACED IS CON-   C
C      SIDERED AS A VECTOR IN N-SPACE, ITS OVER-          C
C      REFLECTION IS OPPOSITE IN DIRECTION, INCREASED IN  C
C      LENGTH BY THE FACTOR 1.3, AND COLLINEAR WITH THE   C
C      REPLACED VERTEX AND CENTROID OF ALL OTHER          C
C      VERTICES.)                                         C
C                                                         C
C      WHEN AN OVER-REFLECTION IS NOT FEASIBLE OR REMAINS C
C      WORST, IT IS CONSIDERED NOT-PERMISSIBLE AND IS     C
C      DISPLACED HALFWAY TOWARD THE CENTROID.  AFTER FOUR C
C      SUCH ATTEMPTS ARE MADE UNSUCCESSFULLY, EVERY FIFTH C
C      ATTEMPT IS MADE BY REFLECTING THE OFFENDING VERTEX C
C      THROUGH THE PRESENT BEST VERTEX, INSTEAD OF        C
C      THROUGH THE CENTROID.  IF 5*N+10 DISPLACEMENTS AND C
C      OVER-REFLECTIONS OCCUR WITHOUT A SUCCESSFUL        C
C      (PERMISSIBLE) RESULT, THE CURRENT BEST VERTEX IS   C
C      TAKEN AS AN INITIAL FEASIBLE POINT FOR A RESTART   C
C      RUN OF THE COMPLETE PROCESS.  RESTARTING IS ALSO   C
C      UNDERTAKEN WHEN 5*NV+10 CONSECUTIVE TRIALS HAVE    C
C      BEEN MADE WITH NO SIGNIFICANT CHANGE IN THE VALUE  C
C      OF THE OBJECTIVE FUNCTION.  IN ALL CASES, RESTART- C
C      ING IS INHIBITED IF THE LAST RESTART DID NOT PRO-  C
C      DUCE A SIGNIFICANT IMPROVEMENT IN THE MINIMUM      C
C      ATTAINED.                                          C
C                                                         C
C      IT IS RECOMMENDED THAT THE USER READ THE REFER-    C
C      ENCE FOR FURTHER USEFUL INFORMATION. IT SHOULD BE  C
C      NOTED THAT THE ALGORITHM DEFINED THERE HAS BEEN    C
C      ALTERED TO FIND THE CONSTRAINED MINIMUM, RATHER    C
C      THAN THE MAXIMUM.                                  C
C                                                         C
C                                                         C
C    REMARKS                                              C
C                                                         C
C      THE INTEGER PROGRAMMING OPTION WAS ADDED TO THIS   C
C      PROGRAM AS SUGGESTED IN REFERENCE (2).  A MIXED    C
C      INTEGER/CONTINUOUS VARIABLE VERSION OF BOXPLX      C
C      WOULD BE EASY TO CREATE BY DECLARING "IP" TO BE AN C
C      ARRAY OF NV CONTROL VARIABLES WHERE IP(I)=1 WOULD  C
C      INDICATE THAT THE I-TH VARIABLE IS TO BE CONFINED  C
C      TO INTEGER VALUES.  EACH STATEMENT OF THE FORM     C
C      'IF (IP .EQ. 1)' ETC. WOULD THEN NEED TO BE AL-    C
C      TERED TO 'IF (IP(I) .EQ. 1)' ETC., WHERE THE SUB-  C
C      SCRIPT IS APPROPRIATELY CHOSEN.  NORMALLY, XU AND  C
C      XL VALUES ARE ALTERED TO BE AN EPSILON 'WITHIN'    C
C      ACTUAL VALUES DECLARED BY THE USER.  THIS ADJUST-  C
C      MENT IS NOT MADE WHEN IP=1.                        C
C                                                         C
C      NOTE:  NO NON-LINEAR PROGRAMMING ALGORITHM CAN     C
C      GUARANTEE THAT THE ANSWER FOUND IS THE GLOBAL      C
C      MINIMUM, RATHER THAN JUST A LOCAL MINIMUM.  HOW-   C
C      EVER, ACCORDING TO REF. 2, THE COMPLEX METHOD HAS  C
C      AN ADVANTAGE IN THAT IT TENDS TO FIND THE GLOBAL   C
C      MINIMUM MORE FREQUENTLY THAN MANY OTHER NON-       C
C      LINEAR PROGRAMMING ALGORITHMS.                     C
```

```
C                                                                              C
C        IT SHOULD BE NOTED THAT THE AUXILIARY VARIABLE                        C
C        FEATURE CAN ALSO BE USED TO DEAL WITH PROBLEMS                        C
C        CONTAINING EQUALITY CONSTRAINTS.  ANY EQUALTIY                        C
C        CONSTRAINT IMPLIES THAT A GIVEN VARIABLE IS NOT                       C
C        TRULY INDEPENDENT.  THEREFORE, IN GENERAL, ONE                        C
C        VARIABLE INVOLVED IN AN EQUALITY CONSTRAINT CAN                       C
C        BE  RENUMBERED FROM THE SET OF NV INDEPENDENT                         C
C        VARIABLES AND ADDED TO THE SET OF NAV AUXILIARY                       C
C        VARIABLES.  THIS USUALLY INVOLVES RENUMBERING                         C
C        THE INDEPENDENT VARIABLES OF THE GIVEN PROBLEM.                       C
C                                                                              C
C     SUBROUTINES AND FUNCTIONS REQUIRED                                       C
C                                                                              C
C        SUBROUTINE 'BOUT' AND FUNCTION 'FBV' ARE INTEGRAL                     C
C        PARTS OF THE BOXPLX PACKAGE.                                          C
C                                                                              C
C        TWO FUNCTIONS MUST BE SUPPLIED BY THE USER.  THE                      C
C        FIRST, KE(X), IS USED TO EVALUATE THE IMPLICIT                        C
C        CONSTRAINTS.  SET KE=0 AT THE BEGINNING OF THE                        C
C        FUNCTION, THEN EVALUATE THE IMPLICIT CONSTRAINTS.                     C
C        IN THE EXAMPLE ABOVE, THE FIRST CONSTRAINT, X(3),                     C
C        MUST BE WITHIN THE RANGE (0. .LE. X(3) .LE. 6).                       C
C        THE SECOND CONSTRAINT X(4), MUST BE .GE. 0. .  IF                     C
C        EITHER CONSTRAINT IS NOT WITHIN THESE BOUNDS,                         C
C        CONTROL IS TRANSFERRED TO STATEMENT 1, AND KE IS                      C
C        SET TO "1" AND CONTROL IS RETURNED TO BOXPLX.                         C
C                                                                              C
C        THE SECOND FUNCTION THE USER MUST PROVIDE EVALU-                      C
C        ATES THE OBJECTIVE FUNCTION.  IT IS CALLED FE(X)                      C
C        AS SHOWN IN THE EXAMPLE ABOVE, AND FE MUST BE SET                     C
C        TO THE VALUE OF THE OBJECTIVE FUNCTION CORRES-                        C
C        PONDING TO CURRENT VALUES OF THE NV INDEPENDENT                       C
C        VARIABLES IN ARRAY 'X'.                                               C
C                                                                              C
C     REFERENCES                                                               C
C                                                                              C
C        BOX.  M.J., A NEW METHOD OF CONSTRAINED OPTIMI-                       C
C        ZATION AND A COMPARISON WITH OTHER METHODS",                          C
C        COMPUTER JOURNAL, 8 APR. '65, PP. 42-52.                              C
C                                                                              C
C        BEVERIDGE G., AND SCHECHTER R., "OPTIMIZATION:                        C
C        THEORY AND PRACTICE", MCGRAW-HILL. 1970.                              C
C                                                                              C
C                                                                              C
C     PROGRAMMER
C
C         R.R. HILLEARY 1/1966.
C         REVISED FOR SYSTEM 360 4/1967
C         CORRECTED  1/1969
C         REVISED/EXTENDED BY L.NOLAN/R.HILLEARY  2/1975
C         CORRECTED  3/1976
C
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8  V(50,50),FUN(50),SUM(25),CEN(25),XS(NV),BU(NV),
     *BL(NV)
C
      KV = 5
      EP = 1.D-6
      NTA = 2000
      IF (NTZ.GT.0) NTA = NTZ
      R = RZ
      IF(R.LE.0.D0.OR.R.GE.1.D0)R=1.D0/3.D0
      NVT = NV+NAV
C
C           TOTAL VARS, EXPLICIT PLUS IMPLICIT
      NT = 0
C           CURRENT TRIAL NO.
      NFT = 0
C           CURRENT NO. OF PERMISSIBLE TRIALS
      NTFS = 0
```

```
C              CURRENT NO. OF TIMES F HAS BEEN ALMOST UNCHANGED
C
C                CHECK FEASIBILITY OF START POINT
C
      DO 4 I=1,NV
      VT = XS(I)
      IF (BL(I).LE.VT) GO TO 1
      II = -I
      VT = BL(I)
      GO TO 2
    1 IF (BU(I).GE.VT) GO TO 3
      II = I
      VT = BU(I)
    2 IF (NPR.GT.0) WRITE (6,49) II
    3 V(I,1) = VT
      CEN(I) = VT
      IF (IP.EQ.1) GO TO 4
      BL(I)=BL(I)+DMAX1(EP,EP*DABS(BL(I)))
      BU(I)=BU(I)-DMAX1(EP,EP*DABS(BU(I)))
    4 SUM(I) = VT
C
C
      NCE = 1
C             NUMBER OF CONSTRAINT EVALUATIONS
      I = 1
      IF (KE(V(1,1)).EQ.0) GO TO 5
      IF (NPR.LE.0) GO TO 12
      WRITE (6,50)
      GO TO 12
    5 NFE = 1
C
C   NUMBER OF VERTICES (K) = 2 TIMES NO. OF VARIABLES.
      K = 2*NV
C
C   NUMBER OF DISPLACEMENTS ALLOWED.
      NLIM = 5*NV+10
C
C
C   NUMBER OF CONSECUTIVE TRIALS WITH UNCHANGED FE TO
C   TERMINATE.
      NCT = NLIM+NV
      ALPHA=1.3D0
      FK = K
      FKM=FK-1.D0
      BETA=ALPHA+1.D0
C
C   INSURE SEED OF RANDOM NUMBER GENERATOR IS ODD.
      IQR = R*1.D7
      IF (MOD(IQR,2).EQ.0) IQR=IQR+101
C
C                          SET UP INITIAL VERTICES
      FUN(1) = FE(V(1,1))
      YMN = FUN(1)
    6 FI = 1.D0
      FUNOLD = FUN(1)
C
      DO 15 I=2,K
      FI = FI+1.D0
      LIMT = 0
    7 LIMT = LIMT+1
C
C   END CALCULATION IF FEASIBLE CENTROID CANNOT BE FOUND.
      IF (LIMT.GE.NLIM) GO TO 11
C
      DO 8 J=1,NV
C
C   RANDOM NUMBER GENERATOR  (RANDU)
      IQR = IQR*65539
      IF (IQR.LT.0) IQR = IQR+2147483647+1
      RQX = IQR
      RQX = RQX*.4656613D-9
      V(J,I) = BL(J)+RQX*(BU(J)-BL(J))
      IF((V(J,I)+.5D0).GT.2147483647.)WRITE(6,100)
```

```
      IF (IP.EQ.1)V(J,I)=IDINT(V(J,I)+.5D0)
    8 CONTINUE
C
      DO 10 L=1,NLIM
      NCE = NCE+1
      IF (KE(V(1,I)).EQ.0) GO TO 13
C
      DO 9 J=1,NV
      VT =      (V(J,I)+CEN(J))*.5D0
      IF((VT+.5D0).GT.2147483647.)WRITE(6,100)
      IF (IP.EQ.1) VT=IDINT (VT+.5D0)
      V(J,I) = VT
    9 CONTINUE
C
   10 CONTINUE
C
   11 IF (NPR.LE.0) GO TO 12
      WRITE (6,51) I
      CALL BOUT (NT,NPT,NFE,NCE,NV,MVT,V,I,FUN,CEN,I)
   12 IER = -1
      GO TO 48
C
   13 DO 14 J=1,NV
      SUM(J) = SUM(J)+V(J,I)
   14 CEN(J)  = SUM(J)/FI
C
C   TRY TO ASSURE FEASIBLE CENTROID FOR STARTING.
      NCE = NCE+1
      IF (KE(CEN).EQ.0) GO TO 60
      SUM(J) = SUM(J) -V(J,I)
      GO TO 7
   60 NFE = NFE+1
      FUN(I) = FE(V(1,I))
   15 CONTINUE
C
C   END OF LOOP SETTING OF INITIAL COMPLEX.
      IF (NPR.LE.0) GO TO 17
      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CEN,0)
C
C   FIND THE WORST VERTEX, THE 'J'TH.
      J = 1
C
      DO 16 I=2,K
      IF (FUN(J).GE.FUN(I)) GO TO 16
      J = I
   16 CONTINUE
C
C   BASIC LOOP. ELIMINATE EACH WORST VERTEX IN TURN. IT
C   MUST BECOME NO LONGER WORST,NOT MERELY IMPROVED. FIND
C   NEXT-TO-WORST VERTEX,THE 'JN'TH ONE.
   17 JN = 1
      IF (J.EQ.1) JN = 2
C
      DO 18 I=1,K
      IF (I.EQ.J) GO TO 18
      IF (FUN(JN).GE.FUN(I)) GO TO 18
      JN = I
   18 CONTINUE
C
C   LIMT=NUMBER OF MOVES DURING THIS TRIAL TOWARD THE
C   CENTROID DUE TO FUNCTION VALUE.
      LIMT = 1
C
C   COMPUTE CENTROID AND OVER REFLECT WORST VERTEX.
C
      DO 19 I=1,NV
      VT = V(I,J)
      SUM(I) = SUM(I)-VT
      CEN(I) = SUM(I)/FKM
      VT = BETA*CEN(I)-ALPHA*VT
      IF((VT+.5D0).GT.2147483647.)WRITE(6,100)
      IF (IP.EQ.1) VT =IDINT(VT+.5D0)
```

```
C
C   INSURE THE EXPLICIT CONSTRAINTS ARE OBSERVED.
 19 V(I,J) = DMAX1(DMIN1(VT,BU(I)),BL(I))
C
      NT = NT+1
C
C   CHECK FOR IMPLICIT CONSTRAINT VIOLATION.
C
 20 DO 25 N=1,NLIM
      NCE = NCE+1
      IF (KE(V(1,J)).EQ.0) GO TO 26
C
C   EVERY 'KV'TH TIME,OVER-REFLECT THE OFFENDING VERTEX
C   THROUGH THE BEST VERTEX.
      IF (MOD(N,KV).NE.0) GO TO 22
      CALL FBV (K,FUN,M)
C
      DO 21 I=1,NV
      VT = BETA*V(I,M)-ALPHA*V(I,J)
      IF((VT+.5D0).GT.2147483647.)WRITE(6,100)
      IF (IP.EQ.1) VT =IDINT(VT+.5D0)
 21 V(I,J) = DMAX1(DMIN1(VT,BU(I)),BL(I))
C
      GO TO 24
C
C   CONSTRAINT VIOLATION:  MOVE NEW POINT TOWARD CENTROID.
C
 22 DO 23 I=1,NV
      VT =      (CEN(I)+V(I,J))*.5D0
      IF((VT+.5D0).GT.2147483647.)WRITE(6,100)
      IF (IP.EQ.1) VT =IDINT(VT+.5D0)
      V(I,J) = VT
 23 CONTINUE
C
 24 NT = NT+1
 25 CONTINUE
C
      IER = 1
C
C   CANNOT GET FEASIBLE VERTEX BY MOVING TOWARD CENTROID,
C   OR BY OVER-REFLECTING THRU THE BEST VERTEX.
      IF (NPR.LE.0) GO TO 42
      WRITE (6,52) NT,J
      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CEN,J)
      GO TO 42
C
C   FEASIBLE VERTEX FOUND, EVALUATE THE OBJECTIVE FUNCTION.
 26 NFE = NFE+1
      FUNTRY = FE(V(1,J))
C
C   TEST TO SEE IF FUNCTION VALUE HAS NOT CHANGED.
      AFO =DABS(FUNTRY-FUNOLD)
      AMX=DMAX1(DABS(EP*FUNOLD),EP)
C
C   ACTIVATE THE FOLLOWING TWO STATEMENTS FOR DIAGNOSTIC
C   PURPOSES ONLY.
C     WRITE(6,99)J,AFO,AMX,FUNTRY,FUNOLD,FUN(J),FUN(JN),
C    *NTFS,N
C 99 FORMAT (1X,I3,6E15.7,2I5)
      IF (AFO.GT.AMX) GO TO 27
      NTFS = NTFS+1
      IF (NTFS.LT.NCT) GO TO 28
      IER = 0
      IF (NPR.LE.0) GO TO 42
      WRITE (6,53) K
      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CEN,0)
      GO TO 42
 27 NTFS = 0
C
C   IS THE NEW VERTEX NO LONGER WORST?
 28 IF (FUNTRY.LT.FUN(JN)) GO TO 34
C
```

```
C     TRIAL VERTEX IS STILL WORST; ADJUST TOWARD CENTROID.
C     EVERY 'KV'TH TIME, OVER-REFLECT THE OFFENDING VERTEX
C     THROUGH THE BEST VERTEX.
         LIMT = LIMT+1
         IF (MOD(LIMT,KV).NE.0) GO TO 30
         CALL FBV (K,FUN,M)
C
         DO 29 I=1,NV
         VT = BETA*V(I,M)-ALPHA*V(I,J)
         IF((VT+.5D0).GT.2147483647.)WRITE(6,100)
         IF (IP.EQ.1) VT =IDINT(VT+.5D0)
   29 V(I,J) = DMAX1(DMIN1(VT,BU(I)),BL(I))
C
         GO TO 32
C
   30 DO 31 I=1,NV
         VT =      (CEN(I)+V(I,J))*.5D0
         IF((VT+.5D0).GT.2147483647.)WRITE(6,100)
         IF (IP.EQ.1) VT =IDINT(VT+.5D0)
         V(I,J) = VT
   31 CONTINUE
C
   32 IF (LIMT.LT.NLIM) GO TO 33
C
C     CANNOT MAKE THE 'J'TH VERTEX NO LONGER WORST BY
C     DISPLACING TOWARD THE CENTROID OR BY OVER-REFLECTING
C     THROUGH THE BEST VERTEX.
         IER = 2
         IF (NPR  .LE. 0)  GO TO 42
         WRITE (6,52)   NT, J
         CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CEN,J)
         GO TO 42
   33 NT = NT+1
         GO TO 20
C
C     SUCCESS:  WE HAVE A REPLACEMENT FOR VERTEX J.
   34 FUN(J) = FUNTRY
         FUNOLD = FUNTRY
         NPT = NPT+1
C
C     EVERY 100'TH PERMISSIBLE TRIAL,RECOMPUTE CENTROID
C     SUMMATION TO AVOID CREEPING ERROR.
         IF (MOD(NPT,100).NE.0) GO TO 37
C
         DO 36 I=1,NV
         SUM(I) = 0.D0
C
         DO 35 N=1,K
   35 SUM(I) = SUM(I)+V(I,N)
C
         CEN(I) = SUM(I)/FK
   36 CONTINUE
C
         LC = 0
         GO TO 39
C
   37 DO 38 I=1,NV
   38 SUM(I) = SUM(I)+V(I,J)
C
         LC = J
C
   39 IF (NPR.LE.0) GO TO 40
         IF (MOD(NPT,NPR).NE.0) GO TO 40
C
         CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,CEN,LC)
C
C     HAS THE MAXIMUM NUMBER OF TRIALS BEEN REACHED WITHOUT
C     CONVERGENCE? IF NOT, GO TO NEW TRIAL.
   40 IF (NT.GE.NTA) GO TO 41
C
C     NEXT-TO-WORST VERTEX NOW BECOMES WORST.
         J = JN
```

```
      GO TO 17
   41 IER = 3
      IF (NPR.GT.0) WRITE (6,54)
C
C    COLLECTOR POINT FOR ALL ENDINGS.
C      1)   CANNOT DEVELOP FEASIBLE VERTEX.                    IER =  1
C      2)   CANNOT DEVELOP A NO-LONGER-WORST VERTEX.           IER =  2
C      3)   FUNCTION VALUE UNCHANGED FOR K TRIALS.             IER =  0
C      4)   LIMIT ON TRIALS REACHED.                           IER =  3
C      5)   CANNOT FIND FEASIBLE VERTEX AT START.              IER = -1
   42 CONTINUE
C
C    FIND BEST VERTEX.
      CALL  FBV (K,FUN,M)
      IF (IER.GE.3) GO TO 44
C
C    RESTART IF THIS SOLUTION IS SIGNIFICANTLY BETTER THAN
C    THE PREVIOUS,OR IF THIS IS THE FIRST TRY.
      IF (NPR.LE.0) GO TO 43
      WRITE (6,55) (M,YMN,FUN(M))
   43 IF (FUN(M).GE.YMN) GO TO 47
      IF(DABS(FUN(M)-YMN).LE.DMAX1(EP,EP*YMN)) GO TO 47
C
C    GIVE IT ANOTHER TRY UNLESS LIMIT ON TRIALS REACHED.
   44 YMN = FUN(M)
      FUN(1) = FUN(M)
C
      DO 45 I =1,NV
      CEN(I) = V(I,M)
      SUM(I) = V(I,M)
   45 V(I,1) = V(I,M)
C
      DO 46 I=1,NVT
   46 XS(I) = V(I,M)
C
      IF (IER.LT.3) GO TO 6
   47 IF (NPR.LE.0) GO TO 48
      CALL BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FUN,V(1,M),-1)
      WRITE (6,56) FUN(M)
   48 RETURN
C
   49 FORMAT('0INDEX AND DIRECTION OF OUTLYING VARIABLE',
     *' AT START',I5)
   50 FORMAT('0IMPLICIT CONSTRAINT VIOLATED AT START.',
     *2X,'DEAD END.')
   51 FORMAT('0CANNOT FIND FEASIBLE',I4,
     *'TH VERTEX OR CENTROID AT START.')
   52 FORMAT('0AT TRIAL ',I4,' CANNOT FIND FEASIBLE VERTEX',
     *'WHICH IS NO LONGER WORST',I4,15X,'RESTART FROM BEST',
     *' VERTEX.')
   53 FORMAT (40H0FUNCTION HAS BEEN ALMOST UNCHANGED FOR I5,
     *7H TRIALS)
   54 FORMAT (27H0LIMIT ON TRIALS EXCEEDED. )
   55 FORMAT('0BEST VERTEX IS NO.',I3,'  OLD MIN WAS ',
     *E15.7,'  NEW MIN IS ',E15.7)
   56 FORMAT ('0MIN OBJECTIVE FUNCTION IS ',E15.7)
  100 FORMAT('0','TRUNCATION ERROR IN BOXPLX')
      END

      SUBROUTINE FBV (K,FUN,M)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 FUN(50)
      M = 1
C
      DO 1 I=2,K
      IF (FUN(M).LE.FUN(I)) GO TO 1
      M = I
    1 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE BOUT (NT,NPT,NFE,NCE,NV,NVT,V,K,FN,C,IK)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 V(50,50),FN(50),C(25)
      WRITE (6,4) NT,NPT,NFE,NCE
C
      DO 1 I=1,K
      WRITE (6,5) FN(I),(V(J,I),J=1,NV)
      IF (NVT.LE.NV) GO TO 1
      NVP = NV+1
      WRITE (6,6) (V(J,I),J=NVP,NVT)
    1 CONTINUE
C
      IF (IK.NE.0) GO TO 2
C
      WRITE (6,7) (C(I),I=1,NV)
      RETURN
    2 IF (IK.GE.0) GO TO 3
      WRITE (6,8) (C(I),I=1,NV)
      RETURN
    3 WRITE (6,9) IK,(C(I),I=1,NV)
      RETURN
C
    4 FORMAT('0NO. TOTAL TRIALS = ',I5,4X,
     *'NO. FEASIBLE TRIALS = ',I5,4X,
     *'NO. FUNCTION EVALUATIONS = ',I5,4X,
     *'NO. CONSTRAINT EVALUATIONS = ',I5/,
     *'0      FUNCTION VALUE',6X,
     *'INDEPENDENT VARIABLES/DEPENDENT OR IMPLICIT',
     *'CONSTRAINTS')
    5 FORMAT (1H ,E18.7,2X,7E14.7/(21X,7E14.7))
    6 FORMAT (21X,7E14.7)
    7 FORMAT (10H0CENTROID 11X,7E14.7/(21X,7E14.7))
    8 FORMAT ('0  BEST VERTEX',7X,7E14.7/(21X,7E14.7))
    9 FORMAT ('0CENTROID LESS VX',I2,2X,7E14.7/(21X,7E14.7))
      END
C                                                                    C
C     A /* CARD GOES HERE                                            C
C                                                                    C
//GO.SYSIN DD *
    1       2       2        1    2      2     1     10
0.0          0.005          15.0
   13      50       2
100.0        0.4            10.0         1.0         0.0         STEP
   14      50       2
16.0         0.6            4.0          1.0         0.0         STEP
18
 1 3 1 3   1100.0           8.0          7.63636363
 2 3 1 4   -400.0           8.0          4.3
 3 3 2 5   -32.0            4.8          7.0
 4 3 2 6   24.0             4.8          5.73333333
 5 1 1 7   100.0
 6 1 2 9   16.0
 7 2 3 7   1.0              0.0
 8 2 4 8   1.0              0.0
 9 2 5 7   1.0              0.0
10 2 6 8   1.0              0.0
11 2 7 9   1.0              7.0
12 2 810   1.0              4.3
13 2 911   1.0              12.0
14 21012   1.0              2.0
15 112 9   2.0
16 11110   4.0
17 111 1   -1.0
18 112 2   -1.0
1.0        0.0              STEP 1
1.0        0.0              STEP 2
MOWREY 10 JAN 1978
THESIS SIMULATION  Y1
MOWREY 10 JAN 1978
THESIS SIMULATION  Y2
```

LIST OF REFERENCES

1. Thaler, G. J., Design of Feedback Systems, Dowden, Hutchison & Ross, Inc. Stroudburg, Pennsylvania, 1973.

2. Ogata, K., Modern Control Engineering, Prenticehall, Inc., Englewood Cliffs, N. J., 1970.

3. Thaler, G. J., and Huang, J., Steady-State Decoupling and Design of Linear Multivariable Systems, Research Report, Grant No. NGR 05-017-010, June 1972-June 1974.

4. Gibson, J. E., Nonlinear Automatic Control, McGraw Hill, New York, 1963.

5. Lima, C. G., Multivariable System Design: A Two Ship Controller for Replenishment at Sea, MSC Thesis, Naval Postgraduate School, Monterey, California, June 1974.

6. MacNamara, M.A.S., A New Optimization Method Applied to Autopilot Design, M.S. Thesis, Naval Postgraduate School, Monterey, California, 1975.

7. Vines, L. P., Computer Automated Design of Systems, M. S. Thesis, Naval Postgraduate School, Monterey, California, 1976.

8. D'Azzo, J. J., and Houpis, C. H., Linear Control System Analysis and Design, McGraw Hill, New York, 1975.

9. Box, M. J., "A New Method of Constrained Optimization and a Comparison with Other Methods," Computer Journal, 8 April 1965, p. 42-52.

10. Beveridge, G., and Schechter, R., Optimization: Theory and Practice, McGraw Hill, New York, 1970.

11. Spendley, W., Hext, G. R., and Himsworth, F. R., Sequential Application of Simplex Designs in Optimization and Evolutionary Design. Technometrics 4, 441-461, 1962.

12. Gill, P. E., and Murray, W., Numerical Methods for Constrained Optimization, Academic Press, Inc., New York, 1974.

13. Syn, W. M., Turner, N. N., Wyman, D. G., DSL/360 Digital Simulation Language User Manual, IBM System/360, November, 1968.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
2.	Department Chairman, Code 62 Naval Postgraduate School Monterey, California 93940	2
3.	Professor George J. Thaler, Code 62Tr Naval Postgraduate School Monterey, California 93940	5
4.	Professor H. A. Titus, Code 62Ts Naval Postgraduate School Monterey, California 93940	1
5.	Major John T. Mowrey 1524 Hidden Brook Drive Herndon, Virginia 22070	3
6.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2